Complemento en WEB

Informática

Diseño y creación de portales web

Gómez, Villar, Alcayde

STARBOOK

ediciones de la U

Informática

Diseño y creación de portales WEB

Julio **Gómez** López
Eugenio **Villar** Fernández
Alfredo **Alcayde** García

STARBOOK
ediciones de la U
conocimiento a su alcance
www.edicionesdelau.com

Diseño y creación de portales web / Julio Gómez López... [et al.]. -- Bogotá : Ediciones de la U, 2011.

242 p. ; 24 cm.
ISBN 978-958-8675-80-0
1. Páginas web estáticas 2. Hojas de estilos 3. Servidores 4. Gestores de contenido I. Tít.
621.39 1a.ed.

Edición original publicada por © StarBook Editorial (España)
Edición autorizada a Ediciones de la U para Colombia

Área: Informática
Primera edición: Bogotá, Colombia, mayo de 2011
ISBN. 978-958-8675-80-0

© Julio Gómez López, Eugenio Villar Fernández, Alfredo Alcayde García
© StarBook. Calle Jarama, 3-A (Polígono Industrial Igarsa) 28860 Paracuellos de Jarama
www.starbook.es / E-mail: edicion@starbook.es
Madrid, España
© Ediciones de la U - Transversal 42 #4B-83 - Tel. (+57-1) 4065861
www.edicionesdelau.com - E-mail: editor@edicionesdelau.com
Bogotá, Colombia

Ediciones de la U es una empresa editorial que, con una visión moderna y estratégica de las tecnologías, desarrolla, promueve, distribuye y comercializa contenidos, herramientas de formación, libros técnicos y profesionales, e-books, e-learning o aprendizaje en línea, realizados por autores con amplia experiencia en las diferentes áreas profesionales e investigativas, para brindar a nuestros usuarios soluciones útiles y prácticas que contribuyan al dominio de sus campos de trabajo y a su mejor desempeño en un mundo global, cambiante y cada vez más competitivo.

Coordinación editorial: Adriana Gutiérrez M.
Carátula: Ediciones de la U
Impresión: Xpress Estudio Gráfico y Digital S.A.
Av.Américas No. 39-53, Pbx. 6020808

Impreso y hecho en Colombia
Printed and made in Colombia

No está permitida la reproducción total o parcial de este libro, ni su tratamiento informático, ni la transmisión de ninguna forma o por cualquier medio, ya sea electrónico, mecánico, por fotocopia, por registro y otros medios, sin el permiso previo y por escrito de los titulares del Copyright.

Apreciad@ lector:

Es gratificante poner en sus manos esta obra, por esta razón le invitamos a que se registre en nuestra web: **www.edicionesdelau.com** y obtenga beneficios adicionales como:

- **Complementos digitales de esta obra**
- **Actualizaciones de esta publicación**
- **Interactuar con los autores a través del blog**
- **Descuentos especiales en próximas compras**
- **Información de nuevas publicaciones de su interés**
- **Noticias y eventos**

Para nosotros es muy importante conocer sus comentarios. No dude en hacernos llegar sus apreciaciones por medio de nuestra web.

Visítenos en www.edicionesdelau.com

aprendiz@je en líne

Desarrollamos y generamos alianzas para la disposición de contenidos en plataformas web que contribuyan de manera eficaz al acceso y apropiación del conocimiento. Contamos con nuestro portal especializado en e-learning:

Visítenos en www.aprendizajeenlinea.com

ÍNDICE

INTRODUCCIÓN **13**

CAPÍTULO 1. PÁGINAS WEB ESTÁTICAS **17**

1.1 INTRODUCCIÓN 17
1.2 ESTRUCTURA BÁSICA 18
1.2.1 Documento HTML 18
1.2.2 Definición del tipo de documento DTD 19
1.2.3 Cabecera de un documento 20
1.2.4 Cuerpo de un documento 22
1.2.5 Mi primera página web 22
1.3 ETIQUETAS BÁSICAS 23
1.3.1 Etiquetas para el formato de texto 23
1.3.2 Comentarios 25
1.3.3 Conjunto de caracteres y caracteres especiales 25
1.3.4 Etiquetas para listas 25
1.3.5 Etiquetas para tablas 26
1.3.6 Colores 28
1.3.7 Imágenes 29
1.3.8 Enlaces 30
1.4 OTROS ELEMENTOS 32
1.4.1 Formularios 32
1.4.2 Frames 35
1.4.3 Eventos 36
1.5 RESUMEN 37

CAPÍTULO 2. HOJAS DE ESTILOS 39
2.1 INTRODUCCIÓN 39
2.1.1 Una breve historia de CSS 40
2.1.2 CSS, HTML, XHTML 41
2.1.3 Utilización 41
2.1.4 El modelo de formato visual y el modelo de caja 44
2.2 CLASES Y PSEUDOCLASES 45
2.2.1 Clases 45
2.2.2 Pseudoclases 46
2.2.3 Herencia y cascada 49
2.3 SELECTORES 50
2.4 PROPIEDADES, VALORES, UNIDADES Y MEDIA-TYPE 53
2.4.1 Propiedades, valores, unidades y media-type 53
2.4.2 Media-types 56
2.4.3 Propiedades de fondo y color 57
2.4.4 Propiedades de textos 59
2.4.5 Propiedades de las cajas 61
2.4.6 Dimensiones en CSS 64
2.5 POSICIONAMIENTO 65
2.5.1 Elementos flotantes 65
2.5.2 Visualización 67
2.5.3 Esquemas (layouts) 69
2.6 RESUMEN 73
2.7 UTILIZACIÓN DE HOJAS DE ESTILO CON ECLIPSE 74
2.7.1 Instalación 74
2.7.2 Utilización del Eclipse Web Developer Tools 76

CAPÍTULO 3. SERVIDORES DE APLICACIONES LAMP 79
3.1 INTRODUCCIÓN 79
3.2 PASOS PREVIOS 80
3.3 APACHE 80
3.3.1 Instalación 81
3.3.2 Configuración 81
3.3.3 Permisos 83
3.4 MYSQL 85
3.4.1 Instalación 85
3.4.2 Tareas básicas 86
3.4.3 La shell de MySQL 86

3.4.4 Interfaces gráficas 90
3.5 PHP 98
3.5.1 MySQL y PHP 100
3.5.2 Modificar el tamaño de subida de ficheros 102

CAPÍTULO 4. PHP 103
4.1 INTRODUCCIÓN 103
4.2 ELEMENTOS BÁSICOS 104
4.2.1 Identificadores 104
4.2.2 Palabras reservadas 105
4.2.3 Comentarios 105
4.2.4 Constantes 106
4.3 TIPOS DE DATOS 106
4.3.1 Tipos de datos escalares 106
4.3.2 Tipos de datos compuestos 107
4.3.3 Tipos de datos especiales 108
4.4 OPERADORES 108
4.4.1 Operadores de asignación 108
4.4.2 Operadores aritméticos 109
4.4.3 Operadores de comparación 110
4.4.4 Operadores lógicos 110
4.4.5 Operadores binarios 111
4.4.6 Operadores de cadena 111
4.5 ESTRUCTURAS DE CONTROL 111
4.5.1 Condicionales simples (if) 112
4.5.2 Selección múltiple 112
4.5.3 Bucles 114
4.5.4 Instrucciones break y continue 115
4.6 FUNCIONES 115
4.6.1 Definición de funciones 116
4.6.2 Funciones predefinidas 116
4.7 VARIABLES PREDEFINIDAS 118
4.8 REUTILIZACIÓN DE CÓDIGO 119
4.9 PHP Y MYSQL 120
4.9.1 Instalación de php-mysql 120
4.9.2 Conexión con el servidor MySQL 120
4.9.3 Seleccionar una base de datos 121
4.9.4 Consultas SQL. 121

4.9.5 Obtener y procesar los resultados 122
4.9.6 Cerrar la conexión con el servidor. 122
4.10 EJEMPLO....122
4.10.1 Creación de la base de datos 124
4.10.2 Conexión a la base de datos....124
4.10.3 Web para mostrar los productos....125
4.10.4 Formulario para insertar....126
4.10.5 Formulario para modificar 128
4.10.6 Formulario para eliminar 131

CAPÍTULO 5. GESTORES DE CONTENIDO 133
5.1 INTRODUCCIÓN 133
5.2 JOOMLA!....138
5.2.1 Requisitos previos....138
5.2.2 Instalación....139
5.2.3 Utilización....145
5.3 MOODLE....154
5.3.1 Requisitos previos....155
5.3.2 Instalación....155
5.3.3 Utilización....163
5.4 MEDIAWIKI 172
5.4.1 Requisitos previos....173
5.4.2 Instalación....173
5.4.3 Personalización....179
5.4.4 Utilización....182
5.5 WORDPRESS....186
5.5.1 Requisitos previos....186
5.5.2 Instalación....187
5.5.3 Utilización....191

CAPÍTULO 6. POSICIONAMIENTO WEB 201
6.1 CONCEPTOS BÁSICOS....201
6.1.1 ¿Qué es un buscador?....202
6.1.2 ¿Qué es una palabra clave? 202
6.1.3 Page rank....204
6.2 POSICIONAMIENTO DE UNA WEB 204
6.2.1 Dar de alta la web en los buscadores....204
6.2.2 Indexación de la web 205
6.2.3 Elección de palabras clave 207

6.2.4 Popularidad de una web 208
6.2.5 Buenas prácticas para mejorar la indexación 208
6.2.6 Cómo detectar y resolver problemas de indexabilidad en la práctica..211
6.2.7 Malas prácticas de posicionamiento 213
6.3 HERRAMIENTAS PARA WEBMASTER 213
6.3.1 Google Analytics 214
6.3.2 Herramientas para webmaster de Google 222
6.3.3 Google AdWords.....224

APÉNDICE I. EL LENGUJAE SQL.....227
I.1 INTRODUCCIÓN 227
I.2 DEFINICIÓN DE DATOS.....229
I.3 MANIPULACIÓN DE DATOS 231

APÉNDICE II. HERRAMIENTAS Y URL REFERENCIADAS.....235

PÁGINA WEB.....239

ÍNDICE ALFABÉTICO.....241

INTRODUCCIÓN

Cada día más y en prácticamente cualquier actividad relacionada con nuestro trabajo, aficiones, para comunicarnos… necesitamos tener acceso a numerosas páginas Web y servicios que pueblan Internet. El número de usuarios de la red de redes no ha dejado de crecer vertiginosamente en los últimos años a medida que evolucionan las tecnologías que la integran.

Buscando información para documentar nuestros estudios, localizando las ofertas de trabajo más recientes, chateando con nuestras amistades en tiempo real o comunicándonos con desconocidos en foros, expresándonos a través de una página personal o difundiendo la imagen de nuestro negocio… nos encontramos en continua interacción con portales Web de cualquier índole.

El objetivo que pretende alcanzar este libro es la introducción del lector al mundo del diseño y creación de portales Web explorando de forma práctica las diferentes herramientas que es necesario utilizar e integrar para conseguirlo. Éstas le permitirán crear sitios Web tanto estáticos como dinámicos, flexibles, eficientes, modernos… garantizando la perdurabilidad del sitio y la optimización tanto del rendimiento como de su propio desarrollo, sea cual sea el propósito del mismo.

Para poder desarrollar y poner en marcha nuestro portal web debemos centrarnos en tres ejes principales.

- **Diseño y creación del portal web.** Para realizar el diseño y la creación de nuestro portal web es necesario la utilización de diferentes tecnologías web. En la figura 0-1 podemos ver de forma gráfica las diferentes tecnologías que nos proporcionan las soluciones más eficientes y ampliamente utilizadas por los diseñadores y desarrolladores Web, y que constituyen gran parte del contenido presentado a lo largo de este libro.

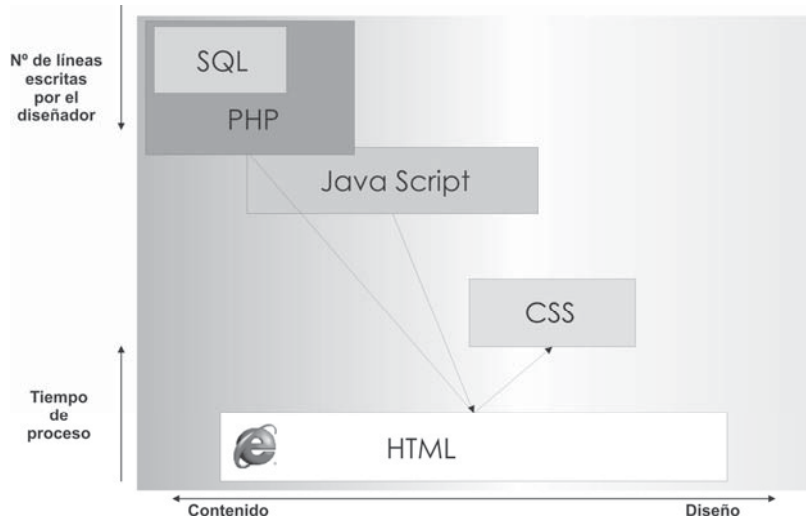


Figura 0-1. Herramientas empleadas para construir un portal

Veamos en qué consiste cada una de estas herramientas y el papel que juega en el proceso de construcción de un portal Web:

- **HTML (Hypertext Markup Language).** Es el lenguaje de marcado para la elaboración de páginas Web. Es utilizado para la descripción de la estructura y el contenido de la página, incluyendo los diferentes elementos que la componen: texto, imágenes, tablas, vídeos, etc. En su sintaxis son utilizadas las denominadas etiquetas, situadas entre corchetes angulares (<, >). Es interpretado por los navegadores Web permitiéndonos visualizar la página Web diseñada.

- **CSS (Cascading Style Sheets).** Las hojas de estilo en cascada son ficheros que permiten definir la forma en que una Web es mostrada en el navegador o impresa. De esta forma indicamos cómo cualquier detalle de la misma es formateado o cuál es su estilo: color, forma, tamaño, fuente, etc. Lo más importante es que permiten separar contenido de presentación, con las ventajas que ello proporciona.

- **JavaScript.** Se trata de un lenguaje de scripting basado en objetos y sin tipos de datos, utilizado para el acceso a objetos en aplicaciones, fundamentalmente integrado en páginas Web, permitiendo generarlas de forma dinámica y con interfaces gráficas mejoradas. JavaScript es interpretado, como HTML, por el navegador Web.

- **PHP (PHP Hypertext Pre-processor).** Es un lenguaje interpretado del lado del servidor que permite la construcción de páginas Web dinámicas. Actualmente es el más extendido de su tipología al ser simple y tener un gran potencial. Uno de sus puntos fuertes es la inclusión de un gran número de funciones para la explotación de bases de datos, junto a SQL, de una forma sencilla.

- **SQL (Standard Query Language).** Es un lenguaje de tipo declarativo para la creación, acceso y manipulación de bases de datos relacionales. Su importancia en el diseño y creación de portales Web radica en las enormes posibilidades para la construcción de consultas sobre bases de datos y obtener información con la que construir páginas Web dinámicamente.

Es interesante apreciar cómo el gráfico anterior refleja el número de líneas escritas por el diseñador y el tiempo empleado para el procesamiento. Mientras que HTML interviene tanto en el diseño como en el contenido de la página y es procesado rápidamente, el número de líneas que debemos escribir es alto. SQL, PHP y prácticamente JavaScript son lenguajes puramente para definir el contenido de la página Web, mientras que con CSS describimos la presentación del mismo. SQL y PHP nos permiten escribir con menos líneas, aunque mayor costo de procesamiento, el código HTML para una página de forma dinámica, al ser ejecutados en el servidor que la ofrece. JavaScript, en cambio, es ejecutado en el cliente que accede a las páginas, es decir, el navegador Web, por lo que su programación debe ser cuidadosa para mantener un buen rendimiento en la visualización de la página.

- **Gestores de contenido.** La demanda constante de información en Internet y la gran rapidez de respuesta a las circunstancias del mercado ha motivado la aparición de métodos y herramientas para el rápido desarrollo y mantenimiento de páginas Web. Separar contenido de diseño en la Web desembocó en la aparición de los gestores de contenido permitiendo crear, mantener y gestionar sitios Web con suma facilidad, y consiguiendo que los administradores puedan centrarse en tareas más creativas. El entorno proporcionado incluye funcionalidades para la creación y gestión de contenido, publicación y presentación, para múltiples usuarios.

De esta forma, mediante el uso de un gestor de contenido, es posible crear y mantener sitios Web dinámicos, en continuo crecimiento, actualizados, flexibles y escalables. Existe un sinfín de gestores de contenido adaptables a cualquiera que sea nuestro propósito: Webs generales, blogs o bitácoras, foros, wikis, portales de docencia virtual o e-learning, comercio electrónico, con fines colaborativos o de trabajo en equipo, y muchos más.

- **Posicionamiento web.** Como hemos citado anteriormente, el impacto de Internet es los modelos de negocio actuales ha provocado que ésta sea la principal vía para publicidad, difusión, venta… en definitiva, de entrar en contacto con clientes y lectores de nuestras páginas Web. Por lo tanto, disponer de un sitio accesible en la Web se ha convertido en prioridad, siendo el medio de mayor relevancia los buscadores: Google, Yahoo!, Bing.

Para finalizar presentamos en este libro los conceptos básicos necesarios (buscadores, palabras clave, PageRank…) para una mejor comprensión del modo de operar del posicionamiento web y cómo mejorarlo. Aprendemos cómo dar de alta nuestro sitio Web en buscadores, cómo es indexada la Web, cómo es medida la popularidad de un sitio, cómo trabajan los robots y spiders, etc.

Para abordar los objetivos que hemos planteado anteriormente se ha estructurado la obra en los siguientes capítulos.

- **Capítulo 1. Páginas Web estáticas.** Introducción; Estructura básica; Etiquetas básicas; Formularios; Frames; Eventos; Resumen.

- **Capítulo 2. Hojas de estilo.** Introducción; Clases y pseudoclases; Selectores; Propiedades, valores, unidades y media-type; Posicionamiento; Otras propiedades CSS; Resumen; Utilización de hojas de estilo con Eclipse.

- **Capítulo 3. Servidores de aplicaciones LAMP.** Introducción; Pasos previos; Apache; MySQL; PHP.

- **Capítulo 4. PHP.** Introducción; Elementos básicos; Tipos de datos; Operadores; Estructuras de control; Funciones; Variables predefinidas; Reutilización de código; PHP y MySQL; Ejemplo.

- **Capítulo 5. Gestores de contenido.** Introducción; Joomla!; Moodle; MediaWiki; WordPress.

- **Capítulo 6. Posicionamiento Web.** Conceptos básicos; Posicionamiento de una Web; Herramientas para webmaster.

Al final del libro se encuentran los siguientes anexos que complementan la obra:

- **Apéndice I. El lenguaje SQL.** Introducción; Definición de datos; Manipulación de datos.

- **Apéndice II. Herramientas y URLs referenciadas.**

Además se pone a disposición del lector el uso de la Web http://www.adminso.es para completar información relacionada con la obra. Tras un proceso de registro, se tendrá acceso a diferente material electrónico como, por ejemplo, presentaciones, ficheros de configuración, etc.

Capítulo 1

PÁGINAS WEB ESTÁTICAS

1.1 INTRODUCCIÓN

Internet se ha convertido en la entidad virtual más variada que ha desarrollado el hombre. El número de usuarios crece periódicamente en cientos de miles por todo el mundo, sin que parezca que vaya a dejar de aumentar. Internet es un lugar virtual donde todo el mundo es bienvenido para hacer negocios, comunicarse, buscar información o, simplemente, divertirse navegando por la red.

En este capítulo vamos a aprender a crear nuestra propia página web utilizando el lenguaje HTML. HTML (**H**yper**T**ext **M**arkup **L**anguage) es un lenguaje de marcas de hipertexto que incorpora al texto etiquetas o marcas que contienen información adicional sobre su estructura o su presentación. HTML es un lenguaje que el navegador va interpretando para representar la página web.

La historia del lenguaje HTML se remonta a 1991 cuando el padre del World Wide Web, Tim Berners-Lee, publicó el primer documento HTML. En su primera versión, el lenguaje poseía 22 etiquetas y era relativamente sencillo. Sin embargo, en la actualidad, la especificación HTML 4.1 está formada por más de 90. A partir de este momento, el lenguaje se ha estado desarrollando y no ha parado de evolucionar.

Para saber más.

En el enlace http://es.wikipedia.org/wiki/Tim_Berners-Lee puedes ver la bibliografía de Tim Bernes-Lee, padre del lenguaje WWW.

En la figura 1-1 se muestra la evolución que ha ido sufriendo el lenguaje HTML desde sus inicios hasta la actualidad.

1989-1991	HTML 1.0 HTML 2.0	Páginas relativamente sencillas
1995	HTML 3.0	Guerra de los navegadors. Los fabricantes competían por tener un navegador con más funciones. El programador debía hacer una página para cada navegador.
1998	HTML 4.0	W3C tomó el mando y puso fin a la guerra de los navegadores creando una sola versión. Se separó estructura de presentación (CSS)
1999	HTML 4.1	Versión más usada, y extendida. Hoy en día se sigue usando y es la más usada.
2000	XHTML 1.0	Unión de XML y HTML. Concepto Web Semántica
2009	XHTML 2.0	Casi ha sido desbancado por HTML 5.0
2009-Futuro	HTML 5.0	Mejor estructura API para video y audio

Figura 1-1. Evolución histórica del lenguaje HTML

En este capítulo vamos a ver la especificación HTML 4.1 por ser la más extendida en la actualidad.

1.2 ESTRUCTURA BÁSICA

Una etiqueta permite indicarle al navegador información sobre los datos que recibe. La sintaxis de una etiqueta es:

```
<etiqueta> Texto </etiqueta>
```

Por ejemplo, como veremos más adelante la etiqueta <b> permite poner el texto en negrita.

```
<b> Texto que aparece en negrita </b>
```

1.2.1 Documento HTML

Básicamente, todo documento HTML tiene la siguiente estructura:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<HTML>
  <head>
  </head>
  <body>
  </body>
</HTML>
```

donde:

- La primera línea contiene información acerca de la versión de HTML con la que estamos trabajando, también conocida como la línea de definición del lenguaje (DTD).
- <HEAD>. Contiene información no visible sobre la página. Esta información puede ser el autor, título, palabras clave para los buscadores, etc.
- <BODY>. Es la parte más importante de la página web y en esta zona es donde se guarda todo el contenido que va a mostrar el navegador.

A continuación vamos a ver de forma resumida los elementos más importantes de cada sección.

1.2.2 Definición del tipo de documento DTD

Para que un documento HTML sea válido según el W3C, debe especificar qué versión de HTML utiliza. Es recomendable utilizar un DTD siempre que sea posible, ya que permite la separación de los datos de la presentación, y al mismo tiempo permite que nuestra página sea fácilmente adaptable a nuevas especificaciones del lenguaje.

HTML, en su especificación 4.01, tiene tres tipos de DTD, de modo que puede seleccionar el que mejor se adapte a nuestras necesidades. A continuación vamos a presentar los tres tipos de DTD desde la versión más restrictiva hacia la más permisiva:

- **DTD Estricto (Strict DTD).** Es la versión más estricta e incluye todos los elementos y atributos de la especificación excepto los elementos y atributos de presentación. W3C no recomienda utilizar los atributos de presentación ya que como veremos más adelante se sustituyen por las hojas de estilo que veremos en el siguiente capítulo.

  La declaración de este tipo de documento se realiza:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
 "http://www.w3.org/TR/html4/strict.dtd">
```

- **DTD Transicional (Transitional DTD).** Incluye todos los elementos del DTD estricto más los elementos de presentación (que W3C no recomienda utilizar).

  La declaración de este tipo de documento se realiza:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
 "http://www.w3.org/TR/html4/loose.dtd">
```

- **DTD para marcos (Frameset DTD).** Es idéntico al DTD transicional con la única diferencia que incluye soporte para frames. Cambiando en el modelo de contenidos del elemento <BODY> por el <FRAMESET>.

La declaración de este tipo de documento se realiza:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
 "http://www.w3.org/TR/html4/frameset.dtd">
```

1.2.3 Cabecera de un documento

La cabecera de un documento HTML se encuentra delimitada por las etiquetas <HEAD> y </HEAD>. En la cabecera se encuentra la información que aunque no se presenta directamente al usuario, es importante. Por ejemplo, se incluye información como: el título de la página, palabras clave (que utilizarán los motores de búsqueda para indexar nuestro documento), información sobre el autor, etc.

Dentro de la cabecera nos podemos encontrar con las siguientes etiquetas:

- **<TITLE> </TITLE>**. Permite indicar el título de la página web que aparecerá en la barra del navegador. Es muy importante asignarle a nuestra web un título, lo más representativo posible. Debido a que esta información será propuesta al usuario, por ejemplo cuando desea archivar la página en favoritos, entonces el título es el nombre que se le presenta para almacenar el enlace a la Web.

```
<HTML>
  <HEAD>
       <TITLE> Título de la página web </TITLE>
  </HEAD>
  <BODY>
    …
  </BODY>
</HTML
```

- **<META>**. Permite especificar metadatos de un documento HTML. Los metadatos son información adicional sobre la página web como el autor, tipo de contenido, etc.

  Los principales atributos de la etiqueta son:

  1. **name**. Identifica un nombre de propiedad.
  2. **content**. Detalla el valor de la propiedad.
  3. **scheme**. Atributo que especifica el esquema que se usará para interpretar el valor de la propiedad.
  4. **http-equiv**. Lo utilizan los servidores HTTP para obtener información sobre los encabezados de respuesta del mensaje HTTP. También puede usarse en lugar del atributo name.

A continuación podemos ver algunos ejemplos de uso de la etiqueta <META>.

```
1. <META name="description" content="Curso Portales Web">
2. <META name="keywords" content="HTML,CSS,JavaScript,PHP">
3. <META name="author" content="Alfredo Alcayde">
4. <META http-equiv="Content-Type"
content="text/html;charset=ISO-8859-1">
```

donde:

- En la línea 1 se introduce una descripción genérica del contenido de nuestro documento HTML.
- En la línea 2, se facilitan las palabras clave que serán utilizadas por los buscadores para indexar nuestra página.
- La línea 3 sirve para aportar información sobre el autor de los contenidos.
- Para terminar, en la línea 4 se especifica la codificación del conjunto de caracteres de nuestro documento.

- **<LINK>.** Este elemento permite crear un vínculo o relación entre nuestro documento y otros recursos. Suele utilizarse para declarar las hojas de estilo que utiliza el documento, el archivo de scripts, describir la posición del documento dentro de una serie, etc. Es importante mencionar que los enlaces realizados con el elemento <LINK>, no se pueden usar para mostrar contenido dentro de un documento.

  A continuación podemos ver un ejemplo:

```
1.<head>
2. <title>Tema 2</title>
3. <link rel="index" href="../indice.html" />
4. <link rel="prev" href="tema1.html" />
5. <link rel="next" href="tema3.html" />
6. <link rel="alternate" media="print" href="tema2-imp.html"/>
7. <link rel="alternate" hreflang="en" href="tema2-en.html"/>
8.</head>
```

  donde:

  - En las líneas 4 y 5 se indica que el documento pertenece a una serie de documentos y que la página anterior es el *tema1.html* y la posterior el *tema3.html*. Por tanto, si el navegador lo permite utilizará esta información para enlazarla con los botones anterior y siguiente de la barra de herramientas.
  - En la línea 6 se da información para que a la hora de imprimir se utilice otro documento que estará formateado para la impresión. Por ejemplo, se habrán eliminado elementos como los menús.
  - En la línea 7 se le avisa al navegador de que existe una traducción al inglés *tema2-en.html* de la página que se muestra.

- **<BASE>.** Permite definir la URL utilizada para resolver todas las direcciones relativas del documento, así como el marco de destino del nuevo enlace. Para ello basta con asignarle la URL al atributo *href* y el marco al atributo *target*.

```
<base href="http://www.ual.es/masterASCI/" target="_blank">
```

  En el ejemplo se toma como dirección base la dirección http://www.ual.es/masterASCI y como marco destino una nueva ventana.

1.2.4 Cuerpo de un documento

El cuerpo de un documento es el lugar de nuestra página web donde se encuentra todo el contenido de nuestra web. Éste se encuentra bajo las etiquetas <BODY> y </BODY> que se encuentran localizadas debajo de la etiqueta </HEAD>. Dentro de esta etiqueta pueden definirse propiedades comunes a toda la página, como color de fondo y márgenes, aunque no es recomendable utilizar los atributos de la etiqueta, ya que se están declarando obsoletos, en favor de las hojas de estilo que veremos en el capítulo siguiente.

1.2.5 Mi primera página web

En este apartado, vamos a construir nuestra primera página web. En el siguiente ejemplo se muestra el código de una página web sencilla, el resultado de la misma lo podemos ver en la figura 1-2. Tal y como hemos visto, la estructura de una página web está compuesta por tres partes bien diferenciadas: la definición del DTD, la cabecera y el cuerpo.

En la primera línea especificamos el DTD del documento, en este caso se ha optado por la versión estricta.

Mientras que en la cabecera, declaramos el título de la página y los metadatos, a través de la cual, proporcionamos información a los buscadores.

Para finalizar, en el cuerpo definimos un título o encabezado <H1>, un párrafo <P> y nuestro primer hiperenlace <A>. Construyendo así, nuestra primera Web.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
          "http://www.w3.org/TR/html4/strict.dtd">
<HTML>
<HEAD>
       <TITLE>Mi primera página Web</TITLE>
       <META name="keywords"content="HTML">
       <META name="description" content="Introducción al HTML">

</HEAD>
<BODY>
       <H1>MI PRIMERA PÁGINA WEB</H1>
       <P>Bienvenidos al mundo del HTML</P>
       <A href="http://www.ual.es/masterACSI" target="_blank">
                    Mi primer enlace: www.ual.es/masterACSI
       </A>
</BODY>
</HTML>
```

Figura 1-2. Mi primera página Web

1.3 ETIQUETAS BÁSICAS

A continuación vamos a ver las etiquetas más utilizadas de HTML.

1.3.1 Etiquetas para el formato de texto

HTML implementa varias etiquetas que nos van a permitir formatear el texto, las cuales podemos ver en la tabla 1-1. Sin embargo, vamos a pasar por alto una de las más utilizadas (<FONT>), que está dejando de ser usada, debido a que hace que el código no sea muy legible.

Tabla 1-1. Etiquetas para formatear el texto

Etiqueta	Descripción
<B>	Texto en negrita
<I>	Texto en cursiva
<U>	Texto subrayado
<DEL>	Texto tachado
<SUP>	Superíndice
<SUB>	Subíndice
<BIG>	Texto grande
<SMALL>	Texto pequeño
<TT>	Texto modo teletipo
<CITE>	Texto modo cita
<DFN>	Texto modo definición
<PRE>	Mantiene el formato del texto, manteniendo espacios y saltos de línea

A continuación vamos a ver un ejemplo básico de la utilización de formato de texto. El resultado de dicho ejemplo lo podemos ver en la figura 1-3. En el ejemplo, hemos destacado en negrita el texto y las etiquetas que afectan al formato, el resto son elementos introducidos para realizar una tabla de presentación de los ejemplos. En los apartados siguientes iremos viendo estas etiquetas.

```
<table border="2" width="78%">
  <tr>
       <td>Formato</td>
       <td>Ejemplo</td>
  </tr>
  <tr>
       <td>Negrita</td>
       <td><b>Texto Formateado</b></td>
  </tr>
  <tr>
       <td>Cursiva</td>
       <td><i>Texto Formateado</i></td>
  </tr>
  <tr>
       <td>Subrayado</td>
       <td><u>Texto Formateado</u></td>
  </tr>
  <tr>
       <td>Tachado</td>
       <td><del>Texto Formateado</del></td>
  </tr>
  <tr>
       <td>Superíndice</td>
       <td>TEXTO<sup>Texto Formateado</sup></td>
  </tr>
   <tr>
       <td>Subíndice</td>
       <td>TEXTO<sub>Texto Formateado</sub></td>
  </tr>
</table>
```

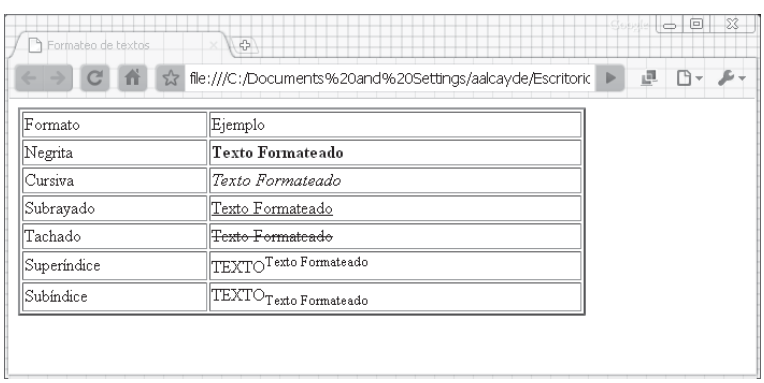


Figura 1-3. Ejemplo de uso de las etiquetas de formato de texto en HTML

1.3.2 Comentarios

HTML, como cualquier otro lenguaje también nos permite agregar comentarios a nuestro código y que éstos no sean mostrados al cliente, de modo que nos faciliten a los programadores la comprensión y reutilización de códigos de otros.

Un comentario en HTML se realiza de la siguiente forma:

```
<!-- Texto del comentario -->
```

1.3.3 Conjunto de caracteres y caracteres especiales

HTML, en su versión 4.01, utiliza la codificación UNICODE UCS4, aunque muchos de los navegadores muestran sólo un pequeño subconjunto de estos caracteres. La información con la que está codificado el conjunto de caracteres la podemos encontrar en el <META>.

Puesto que una página puede ser vista en distintos países, y entre éstos hay diferencias en cuanto a los caracteres utilizados, HTML crea un conjunto denominado "caracteres especiales" que nos permiten identificar de forma inequívoca un caracter. En la tabla 1-2 podemos ver algunos de los más importantes.

Tabla 1-2. Conjunto de caracteres especiales

Etiqueta	Literal	Etiqueta	Literal	Etiqueta	Literal
<	<	&iacoute;	í	¼	¼
>	>	&Oacoute;	Ó	½	½
&	&	&oacoute;	ó	¾	¾
"	"	&Uacoute;	Ú	±	±
	Espacio	&uacoute;	ú	£	£
&Aacoute;	Á	&eacoute;	é	€	€
&aacoute;	á	Ñ	Ñ	¥	¥
&Eacoute;	É	ñ	ñ	¢	¢
&eacoute;	é	©	©	¿	¿
&Iacoute;	Í	®	®		

1.3.4 Etiquetas para listas

Dentro de HTML podemos definir 3 tipos de listas, mediante las cuales podemos estructurar nuestro documento:

- **Listas ordenadas.** Se definen mediante el elemento <OL> y se les añaden los distintos ítems mediante la etiqueta <LI>. En el siguiente ejemplo podemos ver cómo se forman las listas ordenadas.

```
<ol>
 <li> Ítem 1 </li>
 <li> Ítem 2 </li>
</ol>
```

- **Listas no ordenadas**. En este caso se utiliza para definirlas el comando <UL> y para añadir los ítems se utiliza la etiqueta <LI>, la misma que para las listas ordenadas. En el ejemplo podemos comprobar cómo la sintaxis es igual que para las listas ordenadas, ya que sólo cambia el elemento de definición.

```
<ul>
       <li> Ítem 1 </li>
       <li> Ítem 2 </li>
</ul>
```

- **Listas de definiciones**. Son un tipo especial de listas donde cada elemento de la lista está compuesto por un término y una definición y cada una de estas partes tiene su propia etiqueta. Se definen mediante la etiqueta <DL>, el término <DT> y la definición <DD>. En el ejemplo siguiente podemos ver cómo se forma una lista de definiciones.

```
<dl>
       <dt>Término1</dt>
       <dd>Definición1</dd>
       <dt>Término2</dt>
       <dd>Definición2</dd>
</dl>
```

En la figura 1-4, podemos ver el resultado de aplicar los códigos de ejemplo de los tres tipos de lista, mostrando un ejemplo comparativo de los tipos de listas, donde podemos observar las diferencias funcionales entre los distintos tipos.

1.3.5 Etiquetas para tablas

La tabla es uno de los elementos de mayor utilidad debido a su funcionalidad y versatilidad, ya que nos va a permitir organizar y categorizar el contenido dentro de nuestro documento.

Es muy utilizada por los diseñadores para organizar la distribución de los elementos del portal, y aunque es un método bastante utilizado, no es muy recomendable.

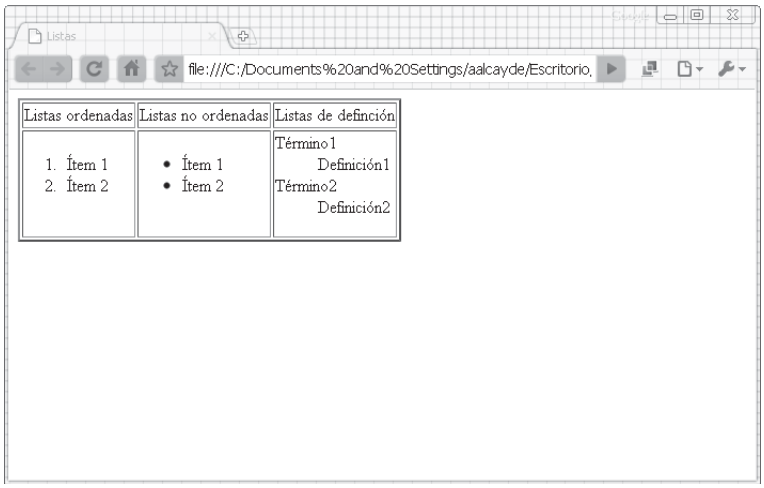


Figura 1-4. Comparación de los distintos tipos de listas en HTML

Los elementos más utilizados para crear una tabla son:

- **<TABLE>.** Se utiliza para definir una tabla y asignarle propiedades generales como por ejemplo: el borde (atributo *border*), la anchura (atributo *width*).
- **<TR>.** Permite definir las distintas filas de la tabla. Debemos tener presente que se debe declarar una etiqueta por cada fila de la tabla. Por tanto, tendremos tantas etiquetas como filas.
- Y dentro de cada fila debemos ir declarando las distintas celdas, para ello se irán añadiendo tantos elementos **<TD>** como celdas se quieran declarar, esta operación habrá que repetirla para cada fila. Para terminar, dentro de cada elemento <TD> incluiremos el contenido de la celda.

En el siguiente ejemplo vamos a construir una tabla sencilla de 3 filas y 3 columnas. Entre comentarios HTML se ha ido explicando la función de cada etiqueta. En la figura 1-5, podemos ver el resultado visual de la construcción de la tabla.

```
<table>                          <!--Definición Tabla-->
  <tr>                           <!--Fila 1-->
    <td>Mes</td>                 <!-- Columna 1, Fila 1-->
    <td>Ingresos</td>            <!-- Columna 2, Fila 1-->
    <td>Gastos</td>              <!-- Columna 3, Fila 1-->
  </tr>                          <!--Fin definición Fila 1-->
  <tr>                           <!--Fila 2-->
    <td>Enero</td>               <!-- Columna 1, Fila 2-->
    <td>1050€</td>               <!-- Columna 2, Fila 2-->
    <td>150€</td>                <!-- Columna 3, Fila 2-->
  </tr>                          <!--Fin definición Fila 2-->
  <tr>                           <!--Fila 2-->
    <td>Febrero</td>             <!-- Columna 1, Fila 3-->
    <td>1150€</td>               <!-- Columna 2, Fila 3-->
```

```
    <td>450€</td>                    <!-- Columna 3, Fila 3-->
  </tr>                              <!--Fin definición Fila 3-->
</table>                             <!--Fin definición Tabla-->
```

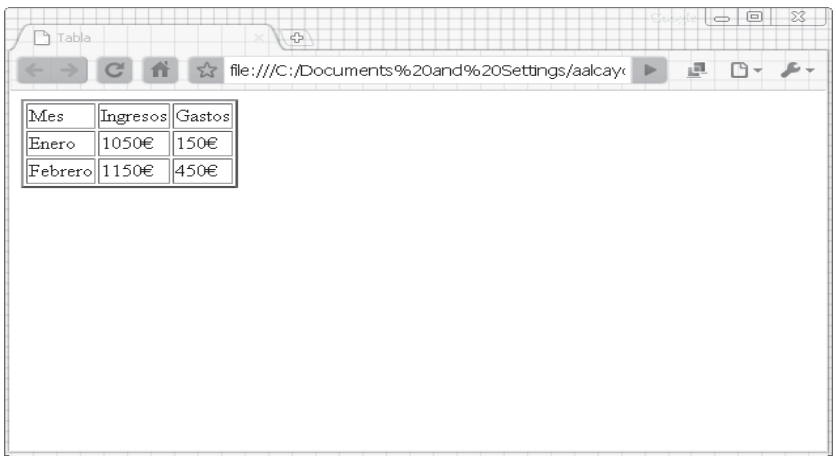


Figura 1-5. Mi primera tabla en HTML

En la tabla 1-3 podemos ver modificadores para las etiquetas de las tablas que nos permiten hacer que una celda ocupe varias columnas (*colspan*) o varias filas (*rowspan*), establecer el tamaño de ancho (*width*) o alto (*height*) de una tabla, o permiten establecer el borde y el color de la tabla.

Tabla 1-3. Otras etiquetas de interés para tablas

Etiqueta	Descripción
<td colspan=2>	Una celda ocupa dos celdas de forma horizontal
<td rowspan=2>	Una celda ocupa dos celdas de forma vertical
<td width=200>	Indica el ancho de una celda, fila o tabla
<td height=200>	Indica el alto de una celda, fila o tabla
<table border=1>	Indica que el borde de la tabla vale 1
<td bgcolor=blue>	Indica el color de una celda, fila o tabla

1.3.6 Colores

Algunas de las etiquetas del lenguaje admiten la propiedad color, bien para dar color al texto, o para establecerlo como fondo del elemento HTML. Para indicar un determinado color se utiliza la combinación de los tres colores básicos (rojo, verde y azul). Para indicar el color podemos utilizar la codificación de 8 bits (valores entre 0 y 255) o la codificación hexadecimal (valores entre 00 y FF).

El color negro en HTML sería: #000000. Adicionalmente HTML define una paleta de 17 colores heredados de la paleta VGA de Windows. Para utilizarlos no haría falta introducir el código del color, tan solo basta con utilizar el nombre.

Tabla 1-4. Paleta de colores HTML

Etiqueta	C. Hexadecimal	Color	Descripción
BLACK	#000000		NEGRO
TEAL	#008080		TEAL
BLUE	#0000FF		AZUL
NAVY	#000080		AZUL MARINO
LIME	#00FF00		LIMA
WHITE	#FFFFFF		BLANCO
PURPLE	#800080		PÚRPURA
YELLOW	#FFFF00		AMARILLO
OLIVE	#808000		OLIVA
RED	#FF0000		ROJO
MAROON	#800000		MARRÓN
GRAY	#808080		GRIS
FUCHSIA	#FF00FF		FUCSIA
GREEN	#008000		VERDE
SILVER	#C0C0C0		PLATA
AQUA	#00FFFF		AGUA

1.3.7 Imágenes

Las imágenes han sido junto con los enlaces, los elementos que más han influido al espectacular desarrollo de la Web, aportando un aspecto visual elegante y una gran interacción entre elementos.

A la hora de insertar imágenes debemos tener en cuenta: que éstas no pueden estar en cualquier formato, sólo se admiten los formatos: JPG, GIF y PNG. Por otro lado, es recomendable adaptar nuestras imágenes en tamaño y resolución para la Web. Bajo la premisa de que sólo va a ser vista en pantalla, por tanto debemos adaptarlas a una resolución estándar. En caso de estar mal adaptadas provocaremos una página de mayor peso que tardará más en cargarse y será menos atractiva para el usuario.

Para insertar una imagen tenemos la etiqueta <IMG>. Para su funcionamiento en este caso, debemos aplicarle una serie de atributos obligatorios.

```
<IMG SRC="URL" ALT="Descripción">
```

donde SRC almacenará la ruta donde está almacenada la imagen y ALT= indicará un texto alternativo que se mostrará al posicionar el cursor del ratón sobre la imagen. Es importante incluir siempre esta información, ya que hará que nuestra Web sea accesible. Por ejemplo para discapacitados, o para que sea más fácilmente indexada en buscadores.

En el ejemplo siguiente: podemos ver el código necesario para incluir una imagen en una Web. Como se comentó anteriormente se han usado los atributos SRC y ALT, en el primero de ellos se proporciona la URL de la imagen a mostrar, en nuestro caso hemos optado por una URL relativa, y en el segundo se proporciona un texto alternativo a la imagen. Para facilitar la accesibilidad.

```
<img src='./img.jpg' alt="Texto descriptivo imagen">
```

La etiqueta *img* permite el uso de varios modificadores entre los que se destacan:

- *border.* Permite indicar el borde de la imagen. Por defecto es 0 pero resulta muy útil establecerlo en el caso de utilizar una imagen como enlace.
- *width* o *height.* Permiten establecer el ancho y alto de la imagen. En el caso de que queramos establecer el tamaño de la imagen lo mejor es establecer únicamente el ancho o el alto, pero no los dos tamaños a la vez para evitar distorsionar el aspecto de la imagen.

En el siguiente ejemplo tenemos una imagen de 100 puntos de ancho y con el borde a 1.

```
<img src='./img.jpg' alt="Texto descriptivo imagen" border=1
with=100>
```

1.3.8 Enlaces

Probablemente el enlace sea el elemento más importante dentro del lenguaje HTML, ya que sin él no tendría sentido hablar de navegación.

Para definir un enlace utilizamos la etiqueta de lenguaje <A></A>, cuya sintaxis es la siguiente:

```
<A HREF="URL" TARGET="Objetivo">Texto del enlace o imagen</A>
```

donde URL será el destino del enlace que puede ser:

- Enlace a un sitio Web distinto al nuestro. Para ello, debemos incluir la URL completa.

  ```
  <a href='http://www.google.es'>Enlace a un sitio web</a>
  ```

- Enlace a páginas de nuestro propio sitio, en este caso podríamos utilizar direccionamiento relativo.

  ```
  <a href='./h1.html'>Enlace a nuestro sitio web</a>
  ```

- Enlace al propio documento, que nos llevaría a marcas dentro de nuestro documento.

```
<a href='#inicio'>Enlace a nuestro documento</a>
```

- Enlace a archivos, que nos permitirá descargarnos el archivo incluido.

```
<a href='archivo.zip'>Descargar fichero</a>
```

- Enlace a direcciones de correo, que permitirá abrir un gestor de correo y enviar un email a la dirección enlazada. Estos enlaces se utilizan cada vez menos debido a que son una fuente para rastrear direcciones a las cuales enviarles mensajes de correo no deseados.

```
<a href='mailto:aalcayde@ual.es'>aalcayde@ual.es</a>
```

La etiqueta TARGET permite indicar el destino donde se abre el enlace, es decir, si se nos muestra en el mismo navegador, en una nueva página del navegador, dentro de un frame, etc. Las opciones más utilizadas son: target=_blank abre una nueva página; target=_top carga la página en la ventana principal; target=nombre carga la página en el frame indicado por su nombre.

Así por ejemplo, si queremos un enlace que cargue en una nueva página la web de Google tendremos:

```
<a href=http://www.google.es target=_blank>Ir a google</a>
```

1.3.8.1 FORMATO DE LAS URL

¿Qué es una URL? El término procede del acrónimo del inglés Uniform Resource Locutor, en castellano, localizador uniforme de recursos. No es más que una cadena de caracteres, bajo un protocolo, utilizada para nombrar cualquier recurso que esté disponible en Internet, para que un usuario cualquiera tenga acceso al recurso.

Para saber más.

RFC para URL
http://tools.ietf.org/html/rfc1738

La sintaxis de una URL se puede descomponer principalmente en 4 partes:

- **Protocolo**. Indica qué servicio estamos utilizando para acceder al recurso, éstos son los típicos servicios de Internet como: www, ftp, telnet, news, etc.
- **Dominio**. Indicará el servidor que aloja el recurso, para ello se hace uso del protocolo IP.
- **Puerto**. Es un parámetro opcional que sólo hay que indicar en el caso de que se haya utilizado un puerto distinto en el servidor para ofrecer el servicio.
- **Ruta al recurso**. Es el camino que hay que seguir para alcanzar al recurso dentro del servidor, para ello hay que indicar la ruta completa.

Dentro del protocolo HTTP, protocolo usado para transmitir las páginas web, las URL se particularizan para poder atender las distintas peculiaridades del protocolo. Por ejemplo, poder resolver las Query Web, que son utilizadas para mandar información del usuario al servidor a través de la URL.

La sintaxis añade dos nuevos elementos a la sintaxis básica que habíamos descrito en el punto anterior quedando como sigue:

```
servicio://dominio:puerto/ruta?parmetro
```

o

```
servicio://dominio:puerto/ruta#enlace
```

La primera forma permite el envío de parámetros que se utilizan para enviar información a otra página. Y la segunda forma permite hacer referencia a un enlace dentro de una página web.

En el siguiente ejemplo podemos ver dos URL válidas para el protocolo HTTP.

```
// URL que envía información al servidor acerca del usuario.
http://masteracsi.ual.es/ficha.html?nombre=alfredo

//URL para acceder al marcador contenidos dentro de la página
objetivos.html
http://masteracsi.ual.es/objetivos.html#contenidos
```

1.4 OTROS ELEMENTOS

1.4.1 Formularios

Los formularios en HTML van a permitir al programador Web recopilar información (datos personales, información de contacto, preferencias…) de los usuarios de la Web, que normalmente es enviada al servidor para su manipulación mediante lenguajes de programación como por ejemplo PHP. Otras veces no es enviada al servidor y es tratada en el cliente mediante lenguajes scripts como JavaScript.

Un formulario básicamente lo podemos dividir en cuatro partes:

1. **Inicio del formulario.** Se define mediante la etiqueta <FORM>, y es en este lugar donde se define la acción del formulario, a través de los atributos METHOD o ACTION. Por ejemplo se indicará la dirección del servidor que va a recibir los datos del formulario.
2. **Cuerpo del formulario.** Es la parte más importante del formulario y es donde se indican los datos que queremos registrar.
3. **Botones de acción**. Permiten enviar o resetear el formulario.
4. **Cierre del formulario**.

En el siguiente ejemplo podemos ver el código para generar el formulario de la figura 1-6, donde podemos apreciar la estructura anteriormente mencionada:

```
<FORM action=http://alqunsitio.com/prog/usuarionuevo method="post">
    Nombre: <INPUT type="text" id="nombre"><BR>
    Apellido: <INPUT type="text" id="apellido"><BR>
    email <INPUT type="text" id="email"><BR>
    <INPUT type="radio" name="sexo" value="Varón"> Varón<BR>
    <INPUT type="radio" name="sexo" value="Mujer"> Mujer<BR>
    <INPUT type="submit" value="Enviar">
    <INPUT type="reset">
</FORM>
```

Las partes del formulario son:

- En la línea 1, se define el inicio del formulario.
- Entre las líneas 2 y 9, se definen los elementos del formulario que capturan los datos introducidos por el usuario.
- En la línea 11 se introduce el botón de envío del formulario y en la 12 el botón de limpieza de los datos introducidos. Los llamados botones de acción.
- Y para terminar en la línea 14, se cierra el formulario.

En la figura 1-6 podemos ver el resultado del formulario anterior.

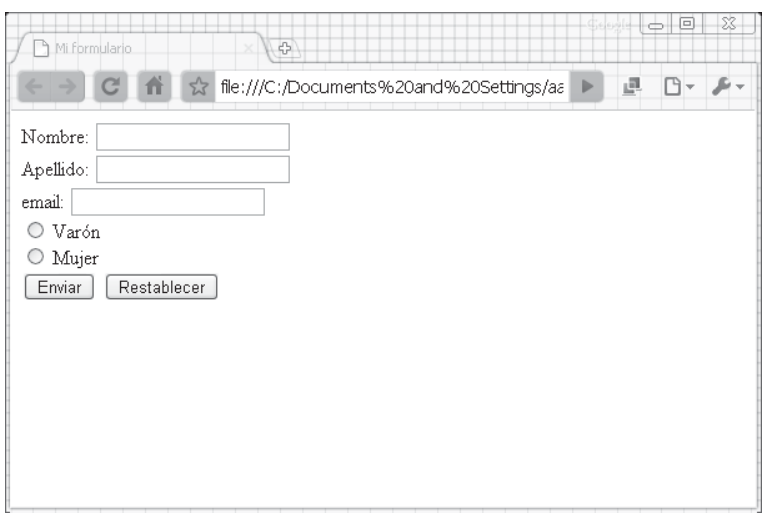


Figura 1-6. Ejemplo de formulario de entrada de datos

Dentro de las etiquetas *<FORM>* y *</FORM>* debemos escribir los diferentes elementos de los que consta el formulario que serán los encargados de recibir los datos del usuario. En la tabla 1-5 podemos ver los diferentes elementos que podemos utilizar en un formulario.

Tabla 1-5. Elementos de un formulario

Elemento	Código	Resultado
Texto corto	Nombre: <Input type=text name=login value="texto por defecto">	Nombre texto por defecto
Password	Clave: <Input type=password name=pass >	Clave: **********
Botones de selección	<input type=checkbox name=var3 checked> opcion 1 	☐ opcion 1
Selección entre varias opciones	<INPUT type=radio name=opcion value=op1> opcion <INPUT type=radio name=opcion value=Op2 checked> opcion 2 NOTA: checked marca la opción por defecto	○ opcion 1 ○ opcion 2
Campos ocultos	<INPUT type=hidden name="variable" value="contraseña">	
Entrada de datos en múltiples líneas	<TEXTAREA name="var7" rows=5 cols=40> Contenido por defecto </TEXTAREA>	Contenido por defecto
Entrada de datos a través de campos de selección	<SELECT name="var8"> <Option value=1> primera opción <Option value=2> segunda opción <Option selected value=3> tercera opcion </SELECT>	Primera opción Primera opción Segunda opción Tercera opción
Botón de envío de datos	<INPUT type=submit value="Enviar datos">	Enviar datos
Botón gráfico de envío de datos	<INPUT type=image src="URL de la imagen" name="variable">	Enviar
Botón de borrado de datos	<INPUT type=reset value="Limpiar formulario">	Limpiar formulario

Es importante indicar que a todos los elementos les tenemos que dar un nombre, utilizando la opción name=nombre, para que la página que recibe los datos del formulario pueda identificar correctamente cada elemento.

1.4.2 Frames

Un frame es un marco o ventana independiente en el que podemos mostrar una página HTML, mediante el cual podemos dividir una página HTML en distintas partes o ventanas, cada una de ellas independientes incluso con sus propias barras de desplazamiento. En la actualidad se están dejando de utilizar estas etiquetas, debido a la usabilidad y accesibilidad, y en la especificación HTML 5.0 desaparecen.

Como etiquetas para el manejo de frames tenemos:

- <FRAMESET> indica al navegador que se va a declarar un conjunto de frames y define sus proporciones en pantalla mediante los atributos ROWS y COLS, indicando los valores en %, o el resto de lo que quede sin utilizar (*).
- <FRAME> define el contenido de la ventana, para ello sólo hay que indicarle en su atributo SRC la página web a visualizar.
- <IFRAME> es uno de los elementos más interesantes dentro de los frames ya que, nos permite crear una ventana en cualquier lugar. Por ejemplo en una celda de una tabla. Su sintaxis sería <IFRAME SRC =”destino” WIDTH=”100%” HEIGHT=”300”>.

A continuación vamos a ver un ejemplo que está dividido en 3 frames. Debemos notar que cuando trabajamos con frames, el cuerpo de la página ya no está formado por la etiqueta <BODY>, ésta deja lugar a la etiqueta <FRAMESET>, etiqueta en la cual definimos la forma en la que estará dividida la pantalla; en este ejemplo hemos optado por una división en 3 columnas, la primera de ellas de un tamaño del 25%, al igual que la última, dejando el resto para la central. Seguidamente se define el enlace con las distintas páginas que deseamos que se vean en cada frame, este proceso se realiza mediante el uso de la etiqueta <FRAME>, a través del atributo SRC se le proporciona la URL de la página. Para finalizar terminaríamos cerrando con la etiqueta </FRAMESET>.

```
<HTML>
<HEAD>
<TITLE>
              Mi primera página con frames
</TITLE>
</HEAD>

<FRAMESET cols="25%,*,25%">
              <frame src="h1.html" />
              <frame src="tabla.html" />
              <frame src="linea.html" />
</FRAMESET >
</HTML>
```

En la figura 1-7, podemos ver el resultado del ejemplo.

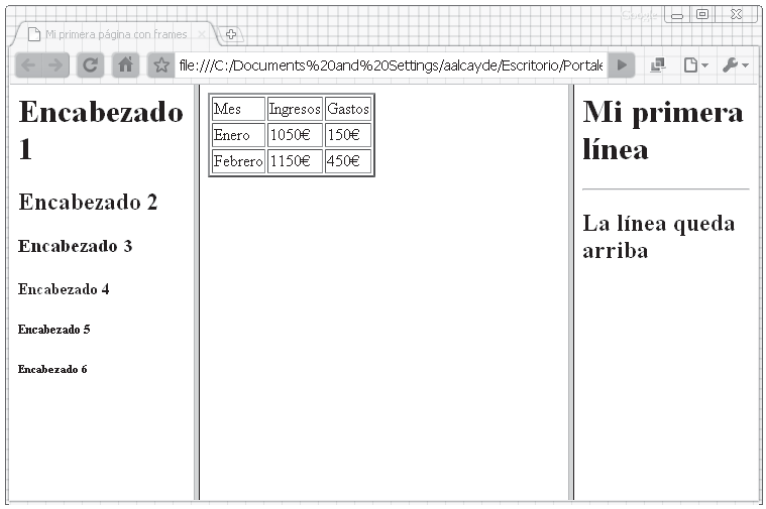


Figura 1-7. Ejemplo de uso de los frames

1.4.3 Eventos

Los eventos que aparecieron por primera vez en HTML 4, añaden a HTML la funcionalidad de interaccionar con el usuario, permitiéndonos por ejemplo: lanzar una llamada a JavaScript para lanzar una rutina.

En la tabla 1-6 podemos ver los elementos que podemos insertar dentro de las etiquetas HTML para generar eventos:

1. **Eventos de ventana**. Sólo válidos para insertar en los elementos <BODY> y <FRAMESET>.
2. **Eventos para formularios**. Como su nombre indica sólo son válidos para elementos de formulario. Y nos permiten registrar las siguientes acciones:
3. **Eventos de teclado**.
4. **Eventos de ratón**.

Tabla 1-6. Eventos

Evento	Descripción
Eventos de ventana	
onLoad	Nos permite lanzar un script a la carga del documento
onUnload	Todo lo contrario, nos permitirá lanzar el script al cierre del documento
Eventos de formulario	
onChange	Detecta cuándo un elemento cambia, por ejemplo cuando insertamos un caracter, o pulsamos sobre un botón
onSubmit	Lanza un script cuando el formulario haya sido enviado
onReset	Nos revela cuándo un formulario ha sido reseteado
onSelect	Ejecuta un script cuando un elemento es seleccionado
onBlur	Averiguamos cuándo el elemento pierde el foco
onFocus	Detectamos cuándo el elemento obtiene el foco
Eventos de teclado	
onKeyDown	Lanza un script cuando una tecla es pulsada
onKeyPress	En este caso cuando la tecla es pulsada y devuelta a su posición
onKeyUp	Lanza el evento al soltar una tecla
Eventos de ratón	
onClick	Lanza el script cuando hacemos un clic de ratón
onDblClick	Lanza el script cuando hacemos doble clic de ratón
onMouseDown	Detecta cuándo el botón del ratón es presionado
onMouseMove	Ejecuta el script cuando movemos el puntero del ratón
onMouseOut	Averiguamos cuándo el puntero del ratón se mueve fuera de la zona de un elemento
onMouseOver	Revela que el puntero del ratón se está desplazando sobre el área de un elemento
onMouseUp	Lanza el script cuando el botón del ratón es liberado

1.5 RESUMEN

Para finalizar, en la tabla 1-7 podemos ver un resumen con las etiquetas HTML más utilizadas.

Tabla 1-7. Etiquetas HTML más utilizadas

Etiqueta	Descripción
<HEAD></HEAD>	Definición cabecera
<BODY></BODY>	Definición del cuerpo
<TITLE> </TITLE>	Título de la página

<!—esto es un comentario -->	Comentarios
<A href="URL a la que accede"> ….. </A>	Creación de enlaces
<IMG src="URL" alt="texto alternativo">	Inserción de imágenes
	Espacio
 	Salto de línea
<P>	Párrafo
<HR>	Línea horizontal
	Negrita
..	Cursiva
<TT> .. </TT>	Teletype
<U> .. </U>	Subrayado
<S>.. </S>	Tachado
^{..}	Superíndice
_{..}	Subíndice
<UL> <LH> Título de la lista </LH> <LI> Elemento 1</LI> <LI> Elemento 2</LI> </UL>	Lista sin numerar
<OL> <LH> Título de la lista </LH> <LI> Elemento 1</LI> <LI> Elemento 2</LI> </OL>	Listas numeradas
TABLE> <TR> <TH> contenido de la celda </TH> <TD> contenido de la celda </TD> </TR>	Tablas
<FRAMESET> <frame src="URL" /> </FRAMESET>	Gestión de frames

Capítulo 2

HOJAS DE ESTILOS

2.1 INTRODUCCIÓN

El estándar HTML se encuentra en continua evolución. El hecho de que Internet se esté abriendo a más público, la competencia entre empresas, y el gran escaparate que supone, provoca en las organizaciones de desarrollo que los navegadores sean cada vez más capaces de plasmar la información de forma atractiva.

Es por ello que aparecen las hojas de estilo. Son una respuesta a la necesidad de organizar la presentación de los elementos de una página Web guardando una armonía, y facilitando el desarrollo de la Web. Las hojas de estilo permiten separar las definiciones de la presentación del contenido de un documento HTML. La especificación HTML fue concebida inicialmente como un lenguaje de marcas, y la introducción de varios niveles de formateo de elementos y atributos de texto sólo sirvió para crear más confusión. Microsoft decidió entonces implementar un nuevo concepto, hojas de estilo, las cuales soportaba en el Internet Explorer 3.0, posteriormente mejoró la especificación en el Internet Explorer 4.0. Netscape también implementó hojas de estilo para sus Communicator y Navigator a partir de la versión 4.0.

Los documentos HTML son fundamentalmente visualizadores de información. Para hacer más atractivo el aspecto de un documento HTML, tenemos a nuestro alcance diversas posibilidades, tales como combinación de colores, distintos tipos de letras, fondo del documento de un determinado color, ubicación en la pantalla de un determinado texto, etc.

En concreto, las hojas de estilo o Style Sheets como se conocen en el mundo Internet, son una extensión del documento en las que se diseña la apariencia general del mismo. La posibilidad de añadir en una hoja de estilo la futura apariencia del documento es un avance para quienes deben programar extensos documentos. Las hojas de estilo permiten indicar el tipo de letra, color, espaciado, fondo, etc.

Las ventajas de la separación de datos-presentaciones con las hojas de estilos son las siguientes:

- **Accesibilidad**. Los datos se pueden presentar en función del usuario y adaptarlos a sus necesidades. No es lo mismo presentar una Web en un PC, que presentarla en un teléfono móvil, o presentarla a personas con deficiencias.
- **Mantenimiento.** Separa y centraliza la presentación, bien en diferentes archivos o en partes bien conocidas del archivo, con lo cual tenemos localizado el código que queremos mantener y/o actualizar.
- **Simplicidad**. Se obtienen documentos más fáciles de entender y se reduce el tamaño de los mismos.

2.1.1 Una breve historia de CSS

Tras la aparición del lenguaje HTML surgieron las hojas de estilo para suplir la necesidad de tener un lenguaje que permita definir de forma estable un formato de presentación o estilo a un conjunto de documentos.

Aunque su verdadero desarrollo se produjo con el boom de Internet y el crecimiento del lenguaje HTML, especialmente en los años de la guerra de los navegadores, ya que la falta de una especificación permitía que una página escrita para un navegador fuera completamente distinta para otro.

En la figura 2-1 podemos ver por orden cronológico la evolución que han sufrido las hojas de estilo a lo largo de los años.

1970	CHSS	Se comienza a hablar de las hojas de estilo. Comienzan a surgir lenguajes de especificación. CHSS (Cascading HTML Style Sheets).
1990	SSP	SSP (Stream-based Style Sheet Proposal).
1994	CSS	Guerra de los navegadores. Se unen los creadores de los lenguajes CHSS y SPP para definir una nueva especificación CSS.
1996	CSS nivel 1	Primera especificación oficial del lenguaje CSS.
1998	CSS nivel 2	Aparece la segunda versión del lenguaje. Sigue siendo utilizada en la actualidad con alguna modificación, CSS 2.1.
2007	CSS nivel 3	Última especificación del lenguaje aunque hoy en día no se ha terminado de implementar en la mayoría de los navegadores.

Figura 2-1. Evolución histórica del lenguaje CSS

Más información

Página oficial de CSS http://www.w3.org/Style/CSS/

2.1.2 CSS, HTML, XHTML

Con la aparición de las hojas de estilo y su creciente popularidad, los diseñadores están cambiando la forma de crear las páginas Web, para adaptarse a los nuevos modelos de diseño.

En una primera etapa, el lenguaje HTML proporciona al diseñador del soporte suficiente para realizar diseños completos, ya que provee de una serie de etiquetas que permiten aplicar estilo al contenido, por ejemplo: la etiqueta <FONT> permite aplicar el tipo de letra, color, tamaño, etc.

Aunque poco a poco las etiquetas que afectan al estilo del documento se están dejando de utilizar y se consideran obsoletas o no recomendadas. Tal y como vimos en el capítulo 1, en la definición 4.1 de HTML, en la versión del tipo de documento más estricto, ya no están permitidos muchos de los elementos que afectan a la presentación.

XHTML lleva al extremo la separación entre la capa de datos y la capa de presentación, eliminando de la especificación aquellos elementos y atributos relacionados con el estilo. XHTML no es nada más que una redefinición del HTML 4 con una sintaxis XML, la cual conserva la mayoría de los elementos y atributos, exceptuando los elementos y atributos relacionados con el estilo, siendo XHTML mucho más estricto en formato del documento. Por ejemplo, obliga a que todos los elementos y atributos estén escritos en minúscula, también que todos los elementos tengan marcas de inicio y fin, etc.

En la actualidad, se tiende a separar los datos de su presentación para poder adaptar mejor el diseño a diferentes dispositivos o usuarios.

Por tanto en una primera fase del diseño, se utiliza un lenguaje como HTML o XHTML, para formatear los contenidos. Es decir, se designa la función de cada elemento dentro de la página: titulares, párrafo, imágenes, tablas...

Una vez se hayan generado los contenidos, se procede a definir, mediante las hojas de estilo (CSS), el aspecto visual de cada elemento: color, tamaño, tipografía, posición dentro de la página, etc.

2.1.3 Utilización

CSS es el lenguaje utilizado para definir el estilo, formato o presentación de documentos separando así los datos o contenido de la presentación, obteniendo en su versión más simple un archivo HTML con el contenido de nuestro documento y otro archivo CSS con la presentación del mismo.

En la figura 2-2, podemos ver dos formas distintas de separar los datos de la presentación. En la primera de ellas, ambas capas se fusionan en el mismo archivo, aunque en partes bien diferenciadas. Mientras que en la segunda, tenemos un archivo, con extensión CSS, donde se define la capa de presentación y un archivo HTML donde se especifican los datos, permitiendo así que el archivo de presentación pueda ser utilizado por más de un archivo de datos.

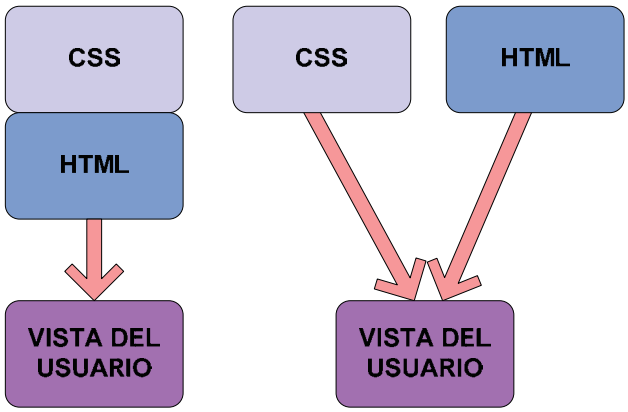


Figura 2-2. Funcionamiento de las hojas de estilo

Un documento CSS se compone de un conjunto de reglas y cada regla consta de las siguientes partes:

- **Selector**, nos indica a qué elemento o parte de la página es aplicable la regla. Normalmente los selectores se corresponden con las etiquetas del lenguaje HTML. Por ejemplo, *table* sería un selector.
- **Declaración**, indica la definición del estilo a aplicar al selector indicado. La declaración a su vez puede dividirse en dos partes:
  - **Propiedad,** que indica el nombre del atributo al que va a ser aplicado.
  - **Valor,** define el nuevo valor de la propiedad.

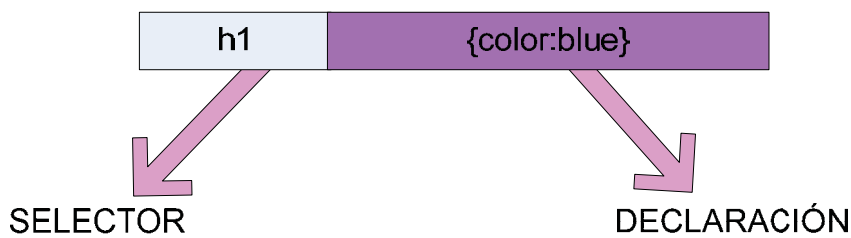


Figura 2-3. Definición de una regla en CSS

Dentro de una regla se puede realizar más de una declaración, para ello las definiciones se separan por punto y coma, no siendo necesario introducirla en la última definición, como podemos ver en el siguiente ejemplo.

```
h1{
	color:blue;
	background-color: red;
	font-size: large;
```

```
      font-family:garamond
}
```

Las reglas de estilo pueden ser declaradas: en línea con el elemento, en la cabecera de un documento, o en un documento externo con extensión CSS. La sintaxis puede verse alterada en función del lugar donde se defina el estilo. En particular difiere si es declarada en línea con el elemento HTML, donde se utilizará el atributo style.

- **Declaración en línea**. La declaración de estilos en línea se utiliza para aplicar el estilo a un elemento en línea con los datos utilizando el atributo *style.* No es recomendable realizar este tipo de declaraciones. Su sintaxis la podemos observar en el ejemplo siguiente:

```
<h1 style = ' color:blue; background-color: red;
  font-size:large; font-family:garamond' > Titulo de la
cabecera
</h1>
```

- **Declaración en la cabecera.** La declaración de estilos en cabecera nos permite la primera separación de datos y presentación de los elementos, ya que, por un lado se declaran los datos, en la cabecera de una página HTML o XHTML, bajo las etiquetas <STYLE> y </STYLE>, tal y como podemos ver en el ejemplo siguiente. Y por otro, los datos en el cuerpo de la página.

```
 <html>
   <head>
<style type="text/css">
<!--
h1{
    color:blue;
    background-color: red;
    font-size: large;
    font-family:garamond
}
-->
</style>
</head>
```

- **Declaración en un documento CSS.** Nos permite separar físicamente los datos de la presentación, en archivos diferentes. Para definir una regla en otro documento ajeno a la página HTML, debemos por un lado crear un documento con extensión .css y seguidamente enlazarlo con nuestra página HTML, tal y como podemos ver en los códigos 2.1 y 2.2.

estilo.css

```
  /*Hoja de estilos*/
h1{
    color:blue;
    background-color: red;
    font-size: large;
    font-family:garamond
}
```

pagina.html

```
<head>
 <link rel="stylesheet" type="text/css" href="estilo.css">
</head>
```

En CSS, como en cualquier otro lenguaje se dispone de la herramienta para introducir comentarios en nuestro código. En CSS los comentarios tienen la misma sintaxis que los del lenguaje C.

```
 /* Comentario*/
```

2.1.4 El modelo de formato visual y el modelo de caja

El modelo de caja es la base de cualquier diseño CSS, ya que todo elemento en CSS se basa en una caja rectangular, ver figura 2-4.

Todo elemento está contenido en una caja que contiene diferentes elementos (áreas o subcajas) que van aportando diferentes opciones al elemento, en lo referente a márgenes, bordes, relleno y contenido del propio elemento.

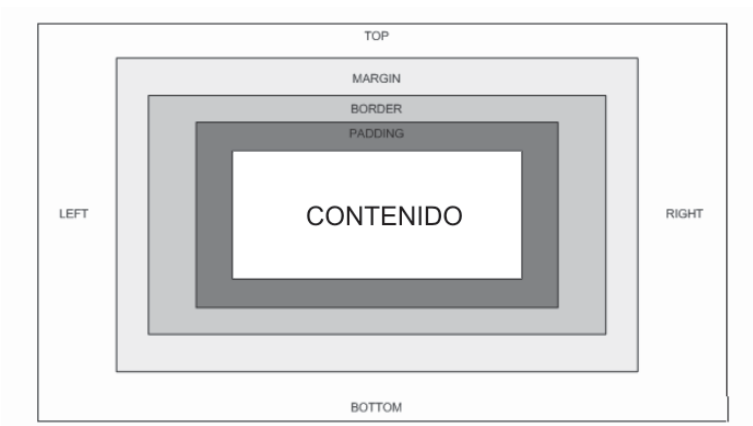


Figura 2-4. Representación gráfica del modelo de caja

A su vez cada una de las distintas cajas: *margin, border y padding,* puede dividirse en los segmentos *left, right, top y bottom*, de forma que se les podrán asignar propiedades diferentes.

Por otro lado, cada caja define su límite, que es el perímetro de cada una de las cuatro áreas. Y cada límite puede dividirse en cuatro segmentos: *right, left, top y bottom*.

Si el límite del *padding* tiene un ancho de 0, el límite del relleno sería el mismo que el límite del contenido. Igual ocurrirá para los límites superiores, por ejemplo si el límite de *margin* tiene un ancho de 0, éste sería igual al límite del borde.

El ancho de la caja está formado por la suma del ancho de los márgenes, bordes, rellenos izquierdos y derechos, y el ancho del contenido. Mientras la altura está dada por la suma de los márgenes, bordes y rellenos superiores e inferiores, y la altura del contenido.

El modelo de formato visual nos permite organizar la distribución de las cajas en pantalla (u otros medios de comunicación visual). El modelo nos proporciona las herramientas para realizar la distribución en pantalla de los distintos elementos.

2.2 CLASES Y PSEUDOCLASES

Para definir los estilos es recomendable utilizar clases o subclases de etiquetas. Además, de las clases que podemos crear nosotros existen pseudoclases asociadas a unas determinadas etiquetas.

2.2.1 Clases

Hasta ahora habíamos definido estilos para una etiqueta determinada. Por ejemplo si se declara:

```
H2 { border-width:2;}
```

todas las frases que vayan entre estas marcas irán enmarcadas por un borde de tamaño 2. Esto puede ser una desventaja si existen determinadas circunstancias en las que no nos interese que dicho selector no vaya enmarcado. Para ello, se provee de un atributo extra en casi todos los elementos del lenguaje HTML, llamado CLASS (clase).

Utilizando este atributo en la marca HTML a la cual uno se refiere, se le puede aislar y se puede referir a él en la hoja de estilo como un subselector.

```
<HTML>
<HEAD>
<STYLE>
P.claseA {
       font-size : 12px;
       font-family : Times New Roman;
       font-weight : bold;
       border-style : solid;
       border-width : 2px;
       text-align : center;
}

</STYLE>
<TITLE>Subselectores en hojas de estilo</TITLE>
</HEAD>
<BODY>
<P>
  Rimas de Gustavo Adolfo B&eacute;cquer
</P>

<P CLASS=claseA>
  Esp&iacute;ritu sin nombre,<BR>
  indefinible esencia,<BR>
```

```
  yo vivo con la vida<BR>
  sin formas de la idea.
</P>

<P CLASS=claseA>
  Yo nado en el vac&iacute;o<BR>
  del sol tiemblo en la hoguera<BR>
  palpito entre las sombras<BR>
  y floto con las tinieblas.
</P>
</BODY>
</HTML>
```

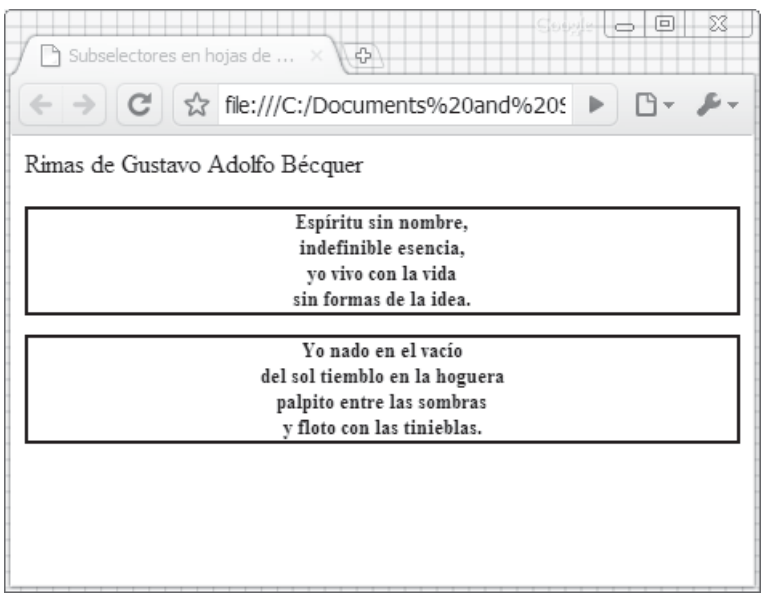


Figura 2-5. Subselectores

En el ejemplo hemos definido una clase, la clase *claseA*, donde indicamos una serie de propiedades a la marca <*P*>. Todas aquellas veces que hagamos uso de la marca <*P*> e indiquemos la propiedad CLASS a *claseA*, inmediatamente adquirirá todas esas propiedades.

2.2.2 Pseudoclases

Diversas pseudo-clases y pseudo-elementos existen en CSS. El estilo normalmente se agrega a un elemento basado simplemente en el tipo de elemento, sin embargo, este modelo no es suficiente para algunos efectos más complejos. Tanto las pseudo-clases y los pseudo-elementos surgieron para paliar estas deficiencias permitiendo dar más dinamismo. Por ejemplo, en HTML no existe un elemento que nos permita seleccionar la primera letra o línea de un párrafo y aplicarle un estilo.

2.2.2.1 PSEUDOCLASES DE VÍNCULO

El ejemplo más usado siempre son los hiperenlaces; uno desearía que los enlaces visitados fueran diferentes de los que no han sido visitados; esto se logra mediante la definición de las siguientes pseudoclases.

```
A: link { color:red}
A: visited {color: blue}
A: active {color: green}
```

Ahora vemos la diferencia, observamos que después del selector, en este caso <A>, aparecen dos puntos (:) ya que después aparece una pseudoclase; mientras que el punto (.) se utiliza para nombrar a clases reales.

En la figura 2-6, podemos ver la aplicación de las pseudo-clases: :*link*, :*visited*, :*active*, y :*hover*. Como podemos observar, el primero de los *enlaces* es de color rojo, porque aún no ha sido visitado mientras que los *enlaces* 2 y 4 han pasado a ser de color azul, atendiendo a que cambiaron de estado al pulsar en ellos y ser visitados, mientras que el enlace 3 es de color negro porque tiene el ratón sobre su marco.

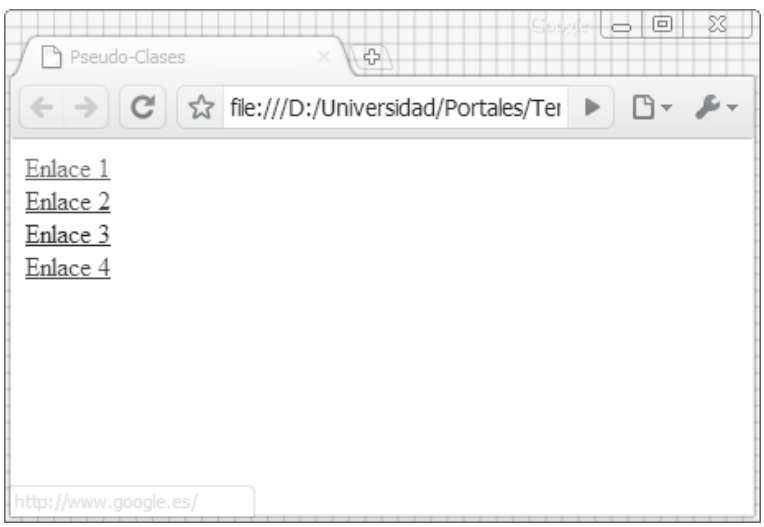


Figura 2-6. Aplicando estilos a hiperenlaces

2.2.2.2 PSEUDO-CLASE DE POSICIÓN :HOVER, :ACTIVE Y :FOCUS

Estas pseudo-clases, aunque originalmente sólo fueron implementadas para el elemento <a>, en la versión CSS2.1 se extendieron al resto de elementos, ya que todo elemento HTML puede estar seleccionado, tener el foco y/o responder al paso del ratón.

Tabla 2-1. Pseudo-clase de posición

Pseudoclase	Descripción
:hover	Selecciona el estilo cuando el cursor del ratón se mueve por el elemento.
:active	Cuando el elemento es seleccionado, por ejemplo cuando se hace clic con el ratón.
:focus	Cuando el elemento recibe el foco.

2.2.2.3 PSEUDOELEMENTOS TIPOGRÁFICOS

Algunos efectos tipográficos comunes están asociados no con elementos estructurales, sino más bien con elementos tipográficos resultantes del formato del documento en el lienzo. En CSS1 se puede hacer referencia a dos de estos elementos tipográficos a través de pseudo-elementos: la primera línea de un elemento, y la primera letra.

El pseudo-elemento *first-line* se utiliza para aplicar estilos especiales a la primera línea de un elemento, una vez que se ha dado formato a éste en el lienzo.

En la figura 2-7, podemos ver un ejemplo visual de uso del pseudo-elemento, viendo cómo va a transformar la primera línea del párrafo.

Este pseudo-elemento nos va a cambiar siempre la primera línea de nuestro párrafo, aportando dinamicidad, porque la línea cambiará en función de la resolución de la pantalla o del tamaño de la venta del explorador.

Figura 2-7. Uso del pseudo-elemento: first-line

El pseudo-elemento *first-letter* se utiliza para aplicar al primer caracter de cada párrafo. Es muy usado para generar las letras capitales, efecto tipográfico muy común.

2.2.3 Herencia y cascada

La herencia y la cascada son dos de los elementos que le dan potencia y sentido al lenguaje CSS, ya que permiten la especificación de las propiedades a los elementos.

2.2.3.1 HERENCIA

Es el método que nos va a permitir que los hijos hereden las propiedades de los padres, según la estructura del documento. Por tanto cuando se especifica una propiedad en un elemento, todos sus descendientes heredan la propiedad de forma inmediata. Por ejemplo, si le especificamos un color a un párrafo, éste se heredará a todos lo elementos que cuelguen del párrafo y permitan la herencia.

Cabe destacar que todas las propiedades no son heredables, por lo que habrá que prestar atención a los elementos que no se heredan. Por ejemplo, no se heredan los elementos relacionados con la definición de márgenes.

En la figura 2-8, cuando aplicamos una propiedad al elemento *<body>* por ejemplo: el tipo de letra, ésta es heredada automáticamente por todos sus descendientes, siempre que no se diga lo contrario, por lo tanto en el elemento *<p>* ya no haría falta definir el tipo de letra.

Sin embargo, si definimos el color de fuente en uno de los elementos *<tr>*, éste es heredado por sus hijos, tal y como vemos en la figura 2-8. Pero no son heredados por el elemento *<tr>* vecino y sus hijos, debido a que se encuentra al mismo nivel. Por tanto, para que se produzca la herencia en un elemento, éste tiene que estar contenido dentro del elemento del que hereda, o lo que es lo mismo ser su hijo.

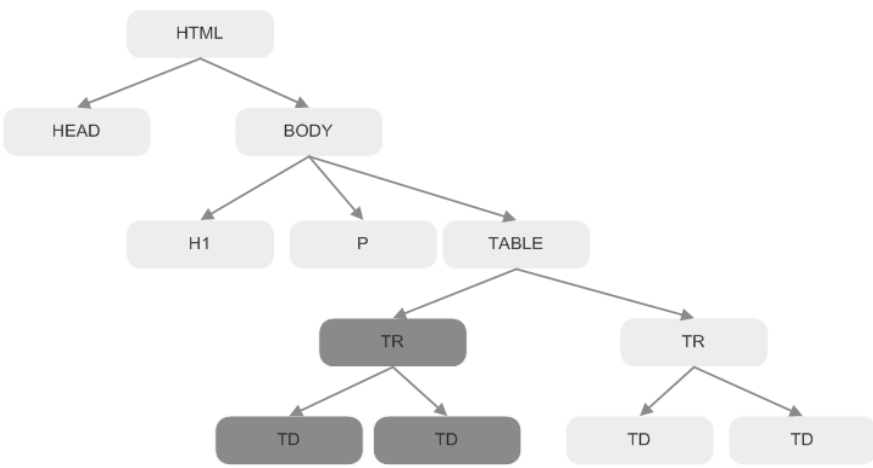


Figura 2-8. Ejemplo de estructura de un documento HTML

2.2.3.2 CASCADA

La cascada es el elemento más importante para aplicar propiedades a un documento, mucho más que la herencia, de ahí que su nombre pase a ser parte del nombre del lenguaje CSS (Hojas de estilo en cascada).

La cascada es el mecanismo que nos permite asignar más de una hoja de estilo a un documento y/o relacionar hojas de estilo entre sí.

Por ejemplo, tenemos una hoja de estilo genérica para nuestro sitio Web, definiendo tipos de letra, colores, etc., por otro lado, tenemos otra hoja para las principales secciones de nuestro sitio, que hará una llamada a la genérica para obtener todas las propiedades definidas y las reutilizaremos o las redefiniremos.

Sin embargo, cuando tenemos varias hojas de estilo aparece un conflicto, podemos tener dos o más reglas a aplicar a un mismo elemento, debiendo decidir qué regla aplicamos. Para solucionar dicho problema, surgen los siguientes conceptos:

- **Especificidad**, es el método por el cual se le asigna un peso o valor a cada regla y la de mayor peso predomina ante la de menor, aplicándose al elemento o documento.
- **Importancia**, es un elemento que se introduce a una regla o definición para aumentar su peso o valor, este elemento se introduce antes del punto y coma introduciendo la palabra clave **!important**. Es decir, ante dos reglas idénticas prevalece la que contiene la palabra clave.
- **Orden**, es la herramienta que nos ayuda a decidir con qué regla nos quedamos ante la igualdad de importancia y especificidad. Ante la igualdad de importancia y especificidad prevalece la última regla encontrada.

Por ejemplo, podemos tener una hoja de estilo base de nuestro sitio Web, que contenga información genérica a la presentación, por otro lado podemos tener otras hojas de estilo para secciones de nuestro sitio y otras más específicas para ciertos contenidos, entre ellas enlazadas dan forma a todo nuestro documento.

2.3 SELECTORES

Los selectores en CSS van a definir tal y como hemos visto en el apartado anterior, el elemento o los elementos a los que aplicar el estilo. Dichos selectores pueden ir desde simples elementos del lenguaje HTML hasta complejos patrones que permitirán aplicar el estilo a un elemento o conjunto de elementos.

- **Selectores de tipo o etiqueta.** Estos selectores son los más sencillos, porque coinciden con las etiquetas de los elementos del lenguaje HTML. Un selector de tipo aplicará el estilo a todos los elementos que coincidan con el mismo tipo.

  Los diseñadores con frecuencia hacen uso del agrupamiento para definir todas las propiedades comunes de los elementos.

- **Selector universal.** El selector universal * es también muy sencillo de utilizar, su función es la de aplicar una propiedad o conjunto de propiedades a todos los elementos de la página.

  No es muy comúnmente utilizado debido a que no es implementado por la mayoría de los navegadores. Y por otro lado, ya que resulta difícil aplicar una misma propiedad a todos los elementos de la página.

- **Selectores de descendientes.** El selector de descendientes es utilizado cuando se desea aplicar una propiedad solamente a un elemento que está contenido dentro de otro elemento. Estos selectores son mucho más específicos que los selectores de tipo.

  Para definir un selector de descendientes se escriben los selectores separados por espacios en blanco seguidos de la definición:

```
Sintaxis:   sel1 sel2 sel3 … selN {Defínción}
```

  En el siguiente ejemplo, sólo se le aplica al elemento *strong* el color *blue*, cuando éste se encuentre contenido dentro de un elemento *span***,** que a su vez se encuentre dentro de un enlace *a***,** y el enlace esté contenido dentro de un elemento *p***.**

```
p a span strong { color: blue }
```

- **Selectores de hijos.** El selector de hijo, aunque muy parecido al selector de descendiente, en el fondo es distinto, ya que sólo se aplica si el descendiente es un hijo directo, en caso contrario, no se aplica.

```
Sintaxis:   sel1>sel2>sel3 >… >selN {Defínción}
```

- **Selector de hermano adyacente.** La funcionalidad del selector adyacente es la de dar formato al elemento hermano inmediatamente siguiente.

```
Sintaxis:   sel1+sel2 {Defínción}
```

  Resulta más fácil entenderlo viendo el resultado gráfico del ejemplo siguiente, que podemos apreciar en la figura 2-9.

```
h1+h2 {
 color:blue;
}
```

  Tal y como podemos observar, se le ha aplicado el color azul al primer elemento *h2*, ya que tiene el mismo padre *body*, y por tanto es hermano del elemento *h1* y además es el siguiente elemento que le precede dentro de la estructura del documento. Sin embargo, no se ha aplicado al *h2* siguiente, porque a pesar de ser hermano, ya no le precede en la estructura del documento.

Figura 2-9. Ejemplo del selector de hermanos adyacentes

- **Selector de atributos.** El selector de atributos nos permite aplicar el formato a un elemento o etiqueta HTML en función de sus atributos y/o valores de los mismos.

  Existen cuatro alternativas de declaración para definir un selector de atributos:

  1. **Elemento [atri].** Aplicará el formato a toda etiqueta de tipo elemento, que tenga un atributo igual a *atrib*.
  2. **Elemento [atrib = val].** En este caso aplicará el formato a todo elemento que tenga al menos un atributo *atrib* y su valor sea *val*.
  3. **Elemento [atrib ~= val].** Selecciona todo elemento, para aplicarle el formato especificado, siempre que posea un atributo *atrib* y al menos uno de sus valores sea *val*.
  4. **Elemento [atrib |= val].** Aplicará el formato a toda etiqueta de tipo elemento, que tenga un atributo *atrib* y su valor empiece por *val* seguido de un guión y cualquier serie de caracteres.

  En el siguiente ejemplo aplicamos el color verde a todo enlace que apunte a la página principal de Google.

```
a[href='http://www.google.es'] {color:green}
```

- **Selector de clases.** El selector de clases, aunque lo vamos a tratar como un selector más, lo podemos definir como un selector de atributos de tipo *elemento[class ~= clase]*, ya que resultaría equivalente.

  Por tanto, el selector de tipo clase nos permite seleccionar todo elemento de tipo *sel* cuyo atributo *class* sea igual a *clase*. Si se obvia el selector *sel*, entonces se selecciona todo elemento que tenga un atributo *class* igual a clase.

```
Sintaxis:   sel.clase{Defínción}
```

- **Selector de ids.** El selector permite a los autores de hojas de estilo seleccionar un elemento por su atributo *id*, si recordamos, el atributo *id* permite asignar un

identificador a una instancia dentro de la estructura del documento. Por tanto, el selector de clases se utiliza normalmente para asignar propiedades específicamente a un solo elemento de nuestra página, ya que como sabemos, el valor de un atributo *id* dentro de una página debe ser único.

```
Sintaxis:   sel#identi{Defínción}
```

A parte del uso de los selectores, CSS permite la combinación de selectores para crear reglas de selección más avanzadas. A continuación veremos algunos ejemplos de posibles agrupamientos de selectores.

```
p.titular a.link titular{color:red}
```

En este primer ejemplo, se selecciona todo elemento de tipo *a* y *class* = *link_titular*, que esté contenido dentro de un párrafo *p* de *class* = *titular*.

```
p.titular  > h2#123  a.link_titular{color:red}
```

En este otro ejemplo, se selecciona todo elemento de tipo *a* y *class* = *link_titular*, que esté contenido en un elemento *h2* con *id='123'*, y además que éste sea hijo de un párrafo de *class* = *titular*.

2.4 PROPIEDADES, VALORES, UNIDADES Y MEDIA-TYPE

2.4.1 Propiedades, valores, unidades y media-type

Las propiedades en CSS permiten especificar valores de medida y color a todos los elementos, de muchas formas distintas. En este apartado conoceremos las distintas formas de aplicar un valor a una propiedad.

2.4.1.1 VALORES

Los valores que puede admitir una propiedad CSS en una hoja de estilo dependen del tipo de propiedad. A continuación se presentan algunos de los valores típicos.

- **Tipos de letra.** El tipo de letra es normalmente el elemento que más quebraderos de cabeza origina a los diseñadores, debido a que los nombres varían en función de las plataformas y puede que existan en unas y en otras no, tomándose por defecto un tipo de letra que no se había previsto y cambiando por completo el diseño de nuestro documento.

  En el ejemplo, se especifica por defecto el tipo de letra *Arial*, en caso de no estar disponible, se utilizará la familia *Helvética* y en último lugar si no estuviera disponible ninguna de las dos, se utilizará la familia *Sans-serif* como fuente base, evitándose así que el navegador seleccione la suya por defecto.

```
P {font-family:''Arial'', ''Helvetica'', sans-serif}
```

- **Medidas y/o longitudes.** Las medidas en CSS se utilizan, entre otras cosas, para especificar la altura, anchura y márgenes de los elementos, y establecer el tamaño de letra del texto. En el siguiente apartado veremos las distintas formas de especificar las medidas.
- **Colores.** Es el valor que permite asignar colores a los distintos elementos. En el siguiente apartado veremos la manera para asignar las unidades de color.

2.4.1.2 UNIDADES

Las unidades de medida se utilizan en varios elementos en las hojas de estilo como es el caso del tamaño de la letra, borde, etc. Las unidades de medida las podemos definir de tres formas posibles:

- **Absoluta.** Las unidades de longitud absolutas tienen el problema de que están muy ligadas con el medio de salida y nos crean problemas de accesibilidad. En la tabla 2-2 se muestran las diferentes unidades de medida absolutas.
- **Relativa.** Las unidades relativas son más flexibles que las unidades absolutas porque se adaptan más fácilmente a los diferentes medios, ya que nos permiten definir la medida en función del dispositivo donde se muestre. Por ejemplo, en el siguiente ejemplo definimos el tamaño de letra con un valor de medida de + 3 píxels.

  ```
  P {font-size: +3px}
  ```

  En la tabla 2-3 se muestran las diferentes unidades de medida relativas.
- **Porcentajes.** Aparte de las unidades relativas, CSS incluye otra forma de medida relativa basada en porcentajes. En este caso se forma añadiendo un número seguido del símbolo % sin espacios entre ellos.

  En el siguiente ejemplo, se define el tipo de letra a un 200% del tipo de letra estándar.

  ```
  p{font-size:200%}
  ```

  Por lo general, se recomienda el uso de unidades relativas siempre que sea posible, ya que mejora la accesibilidad de la página y permite que los documentos se adapten fácilmente a cualquier medio y dispositivo.

- **Unidades de color.** Permiten indicar el color que queremos utilizar y que veremos en el siguiente apartado.

Tabla 2-2. Unidades de medida absolutas

Abreviatura	Unidad	Equivalencia
Pm	Pulgadas	2,54 cm
Cm	Centímetros	
Mm	Milímetros	
Pt	Puntos	$1/72$ pulgadas
Pc	Picas	12 puntos

Tabla 2-3. Unidades de medida relativas

Abreviatura	Unidad
Em	Da el valor en función de la anchura de la letra M mayúscula.
Ex	En el caso de la Ex ésta asigna el valor en función de la altura de una x minúscula.
Px	Píxel asigna el tamaño en función de la resolución del dispositivo donde se presenta.

2.4.1.3 COLORES

En CSS se definen unos 17 colores que son heredados de la paleta VGA de Windows, y aún se sigue manteniendo en la actualidad, por mantener compatibilidades. En la tabla 2-4 podemos ver los colores predefinidos.

Cabe mencionar que este método no es utilizado, porque aunque es muy fácil de recordar, tiene una gama de colores muy reducida.

Tabla 2-4. Paleta de colores CSS

Etiqueta	C. Hexadecimal	Color	Descripción
BLACK	#000000		NEGRO
TEAL	#008080		TEAL
BLUE	#0000FF		AZUL
NAVY	#000080		AZUL MARINO
LIME	#00FF00		LIMA
WHITE	#FFFFFF		BLANCO
PURPLE	#800080		PÚRPURA
YELLOW	#FFFF00		AMARILLO
OLIVE	#808000		OLIVA
RED	#FF0000		ROJO
MAROON	#800000		MARRÓN
GRAY	#808080		GRIS
FUCHSIA	#FF00FF		FÚCSIA
GREEN	#008000		VERDE
SILVER	#C0C0C0		PLATA
AQUA	#00FFFF		AGUA

Los colores se pueden establecer de las siguientes formas:

- **RGB decimal.** Utiliza el modelo RGB para especificar un color. Se crea especificando la cantidad de rojo, verde y azul que contiene el color, en el caso de decimal especificaremos la proporción de los tres colores entre 0 y 255.

Por ejemplo el color negro se definiría así:

```
P {color:rgb(0,0,0))
```

- **RGB hexadecimal.** En este caso se utiliza la notación hexadecimal, con lo cual especificamos la proporción del color con un valor entre 00 y FF. Esta notación es más fácil de recordar.

  Por ejemplo el color negro en hexadecimal se definiría así:

```
   P {color:#000000)
```

- **RGB porcentual.** En este caso, la proporción de colores básicos del modelo RGB se especifica mediante porcentajes de rojo, verde y azul.

  Por ejemplo el color negro se definiría así:

```
   P {color:rgb(0%,0%,0%))
```

Para saber más.

En los siguientes enlaces podemos ver los colores que se recomienda utilizar para el diseño de páginas web:
http://en.wikipedia.org/wiki/Web_colors#Web-safe_colors
http://www.colourlovers.com/

2.4.2 Media-types

Los media-types nos permiten especificar cómo se va a mostrar nuestra Web dependiendo del medio donde se presente. No será lo mismo mostrar nuestra Web en la pantalla de un ordenador, que en un dispositivo móvil o en una impresora.

Para ello se hace uso de la palabra clave @media y su sintaxis es:

```
@media  dispositivo
{
       Definición de estilos para el dispositivo.
}
```

En la tabla 2-5 podemos ver los diferentes media-types que define el estándar CSS2.1.

Tabla 2-5. Media-types

Media-type	Descripción
all	Todos los medios
aural	Sintetizadores de voz
braille	Dispositivos braille
embossed	Impresoras braille
handheld	Dispositivos con pequeñas pantallas: PDA, móviles…
print	Impresoras
projection	Para presentaciones
screen	Pantallas de ordenadores
tty	Impresoras con puntos
tv	Para televisores

2.4.3 Propiedades de fondo y color

Estas propiedades permiten establecer el color (a menudo llamado en inglés foreground color) y el fondo (background) de un elemento. Se puede establecer un color de fondo y/o una imagen de fondo.

También se puede establecer la posición de la imagen, si la imagen se repite y cómo se repite, y si está fija o si se mueve con respecto al lienzo (es decir, si hace scroll).

La propiedad *color* se hereda normalmente y las propiedades de fondo no se heredan, pero el fondo del elemento padre se verá por defecto a través de los hijos, ya que el valor inicial de *background-color* es transparente.

A continuación vamos a ver las propiedades más importantes:

- *color.* Permite especificar el color del texto de un elemento. Tal y como hemos visto anteriormente hay distintas maneras de especificar el color. Veamos varios ejemplos:

```
EM { color: red }              /* lenguaje natural */
EM { color: rgb(255,0,0) }     /* RGB con rango 0-255 */
```

- *background-color.* Permite indicar el color del fondo de un elemento.

```
H1 { background-color: #FF0000 }
```

- *background-image.* Permite indicar que vamos a utilizar una imagen como fondo de un elemento. Cuando se establece una imagen de fondo, también debería establecerse un color de fondo que será usado en el caso de que la imagen no esté disponible. Si la imagen está disponible, se superpone al color de fondo.

```
BODY { background-image: url(marmol.gif) }
P { background-image: none }
```

- *background-repeat.* Se utiliza cuando se establece una imagen como fondo y permite indicar el modo en que se repite la imagen. Un valor de *repeat* significa que la imagen se repite tanto horizontal como verticalmente. Los valores *repeat-x* y *repeat-y* hacen que la imagen se repita horizontalmente o verticalmente creando una sola banda de imágenes de un lado a otro (de arriba a abajo). Con un valor 'no-repeat' la imagen no se repite.

```
BODY {
     background: red url(pendiente.gif);
     background-repeat: repeat-y;
}
```

- *background-attachment.* Si se especifica una imagen de fondo, el valor de *background-attachment* determina si está fija con relación al lienzo, o si se mueve (hace scroll) junto con el contenido. En el ejemplo siguiente la imagen queda fijada.

```
BODY {
    background: red url(pendiente.gif);
    background-repeat: repeat-y;
    background-attachment: fixed;
}
```

- *background-position.* Si se ha especificado una imagen de fondo, el valor de *background-position* permite indicar la posición de la imagen. Por ejemplo, los valores '0% 0%' sitúan la imagen en la esquina superior izquierda; los valores '100% 100%' sitúan la imagen en la esquina inferior derecha; los valores '14% 84%' sitúan la imagen al 14% de la izquierda y al 84% de su parte superior; los valores '2cm 2cm' sitúan la imagen a 2 cm a la derecha y 2 cm por debajo de la esquina superior izquierda del elemento.

  Si sólo se da un valor porcentual o una longitud, este valor sólo establece la posición horizontal, y la posición vertical será del 50%.

  También se pueden usar como valores palabras clave que indican la posición de la imagen de fondo. Las palabras clave no pueden combinarse con valores porcentuales ni con longitudes. A continuación vamos a ver unos ejemplos de uso:

```
BODY { background: url(banner.jpeg) right top }    /* 100%
0% */
BODY { background: url(banner.jpeg) top center }   /*  50%
0% */
BODY { background: url(banner.jpeg) center }       /*  50%
50% */
BODY { background: url(banner.jpeg) bottom }       /*  50%
100% */
```

  Si la imagen de fondo está fija con relación al lienzo (véase la propiedad *background-attachment*), la imagen se coloca con relación al lienzo y no con relación al elemento. En el siguiente ejemplo, la imagen se coloca en la esquina inferior derecha del lienzo.

```
BODY {
     background-image: url(logo.png);
     background-attachment: fixed;
     background-position: 100% 100%;
}
```

- *background.* La propiedad *background* es una propiedad abreviada para establecer las propiedades de fondo individuales (es decir, *background-color*, *background-image*, *background-repeat*, *background-attachment* y *background-position*) en un mismo lugar de una hoja de estilo.

  Los posibles valores de las propiedades *background* son el conjunto de todos los posibles para cada una de las propiedades individuales.

  En el ejemplo de la figura 2-10, podemos ver el resultado gráfico de la aplicación de las reglas CSS para los elementos *BODY* y *P* con sus imágenes de fondo.

```
<html>
 <head>
    <style>
       body{background:url(logo.jpg) gray 0% 0% no-repeat
fixed;}
       P{background:url(logoMirada.jpg) no-repeat;}
    </style>
    </head>
    <body>
       <br>
       <p>
            Parrafo <br><br><br><br><br><br><br><br><br><br><br>
       </p>
    </body>
</html>
```

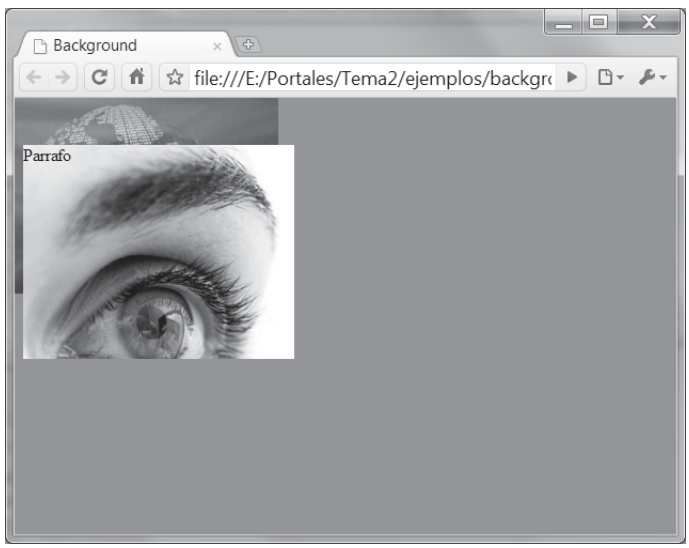


Figura 2-10. Ejemplo de uso del atributo background

2.4.4 Propiedades de textos

Estas propiedades permiten dar formato al texto de un elemento. Se puede establecer por ejemplo atributos como la alineación, decoración, tamaño, etc.

A continuación vamos a ver las propiedades más importantes:

- *text-decoration.* Describe las decoraciones que se añaden al texto de un elemento. Si el elemento no tiene texto (por ejemplo, el elemento <*img*> en HTML) o es un elemento vacío (p.e. <*em*></*em*>) esta propiedad no tiene efecto.

  El color o colores requeridos por la decoración del texto deberían derivar del valor de la propiedad *color*.

Esta propiedad puede tomar como atributos los valores: *none*, *underline*, *overline*, *line-through* y *blink*, debe ser reconocida por los navegadores, pero puede que no sea implementada. Por ejemplo, Internet Explorer no soporta el efecto.

```
A:link, A:visited, A:active { text-decoration: underline }
```

En el ejemplo, los textos de todos los vínculos aparecerán subrayados.

- *text-transform.* Permite aplicar transformaciones al texto de un elemento. Por ejemplo permite transformar el texto a mayúsculas o minúsculas.

  Esta propiedad puede tomar como atributos los valores: *none*, *capitaliza*, *uppercase*, *lowercase*.

```
H1 { text-transform: uppercase }
```

El ejemplo convertirá el texto de todos los elementos 'H1' en mayúsculas, independientemente de cómo se hayan escrito los datos.

- *text-align.* Esta propiedad indica cómo se alinea el texto dentro del elemento. Esta propiedad puede tomar como atributos los valores: *left*, *right*, *center* y *justify*.

```
DIV.center { text-align: center }
```

Como *text-align* se hereda, todos los elementos en bloque dentro del elemento *DIV* con *CLASS=center* estarán centrados. Obsérvese que las alineaciones son relativas a la anchura del elemento, no del lienzo. Si *justify* no está soportado, se sustituirá por otro estilo que normalmente es *left*.

- *text-indent.* Esta propiedad especifica la sangría de la primera línea formateada. El valor de *text-indent* puede ser negativo, pero puede haber limitaciones específicas de cada implementación.

  No se inserta sangría en medio de un elemento que haya sido roto por otro (por ejemplo, con la etiqueta <*br*> en HTML).

  A continuación, podemos ver algunos ejemplos de uso de la propiedad.

```
P { text-indent: 3em }
P.pixel{ text-indent: 12px }
P.porce{text-ident: 5%}
```

- *word-spacing.* Permite especificar el espacio por defecto entre palabras. Los valores pueden ser negativos.

```
H1 { word-spacing: 1em }
```

En el ejemplo, el espaciado de palabras entre cada palabra contenida en elementos <*H1*> se incrementa en *1em*.

- *letter-spacing.* Permite modificar el espacio por defecto entre caracteres. Con un valor *normal*, el navegador puede modificar el espacio entre letras para justificar el

texto. Esto no sucederá si *letter-spacing* está establecido explícitamente a un valor de tipo <longitud>.

```
BLOCKQUOTE { letter-spacing: 0.1em }
```

En el ejemplo anterior, el espaciado de letras entre cada caracter de los elementos *BLOCKQUOTE* se incrementa en *0.1em*.

- *line-height.* Esta propiedad establece la distancia entre las líneas de base de dos líneas adyacentes.

  Un valor *normal* establece la altura de línea a un valor razonable para la fuente del elemento. No se permiten valores negativos.

  En el ejemplo, las tres reglas dan el mismo resultado de altura de línea.

```
DIV { line-height: 1.2; font-size: 10pt }    /* número */
DIV { line-height: 1.2em; font-size: 10pt }  /* longitud */
DIV { line-height: 120%; font-size: 10pt }   /* porcentaje */
```

- *vertical-align.* Permite modificar la alineación vertical del elemento.

  Esta propiedad puede tomar como atributos los valores: *baseline*, *sub*, *super*, *top*, *text-top middle*, *bottom text-bottom*.

2.4.5 Propiedades de las cajas

En este apartado se van a describir los distintos atributos que permiten manipular y dar formato a las cajas.

A continuación vamos a ver las propiedades más importantes:

- *margin.* Establece el margen de un elemento. Se puede utilizar de forma abreviada especificando el margen para cada uno de los lados de la caja.

  Si se especifican cuatro valores de longitud, se aplican a los márgenes superior, derecho, inferior e izquierdo respectivamente. Si sólo hay un valor, se aplica a todos los lados; si hay dos o tres valores, se toma para los valores que faltan los del lado opuesto.

  A continuación podemos ver un ejemplo de utilización:

```
BODY {margin: 2em } /* todos los márgenes puestos a 2em */
BODY {margin: 1em 2em } /* top y bottom=1em,
                           right y left=2em */
BODY {margin: 1em 2em 3em } /* top=1em, right=2em,
                          bottom=3em, left=2em */
```

La última regla del ejemplo equivale al conjunto de reglas del siguiente ejemplo, donde se especifican uno a uno los márgenes de cada uno de los lados de la caja.

```
BODY {
  margin-top: 1em;
  margin-right: 2em;
  margin-bottom: 3em;
  margin-left: 2em;
}
```

- *padding.* Permite establecer el espacio entre el borde y el contenido. Se comporta de la misma forma que la propiedad margin.

```
H1 {
    background: white;
    padding: 1em 2em;
}
```

En el ejemplo anterior, se establece en *1em* el relleno vertical (*padding-top* y *padding-bottom*), y en *2em* el horizontal (*padding-right* y *padding-left*). La unidad *em* es relativa al tamaño de fuente del elemento: *1em* es igual al tamaño de la fuente en uso. La superficie del área de relleno la establecemos con la propiedad *background*.

- *border.* Permite establecer el borde de un elemento, definiendo, anchura, color y estilo. La propiedad *border* es una propiedad abreviada para establecer la misma anchura, color y estilo en los cuatro bordes de un elemento.

Para establecer estas propiedades por separado tenemos las propiedades *border-style*, *border-color* y *borde-width*, que se comportan de forma análoga a las propiedades *padding* y *margin*.

Las reglas del siguiente ejemplo serían equivalentes.

```
P {
    border-top: solid red;
    border-right: solid red;
    border-bottom: solid red;
    border-left: solid red
}
```

```
P { border: solid red }
```

A diferencia de las propiedades abreviadas *margin* y *padding*, la propiedad *border* no puede establecer valores distintos para cada borde. Para ello debe utilizarse una o más de las otras propiedades de borde.

En el siguiente ejemplo, el borde será una línea continua (*solid*) negra.

```
P {
    color: black;
    background: white;
    border: solid;
}
```

En este ejemplo, los bordes horizontales serán continuos (*solid*) y los bordes verticales punteados (*dotted*).

```
#xy34 { border-style: solid dotted }
```

- *outline*. Las propiedades de outline establecen un perfil sobre los elementos. Aunque ésta es muy parecida a la propiedad borde difiere en diferentes aspectos: los perfiles (outline) no ocupan espacio, ya que se dibujan sobre los elementos, a diferencia de los bordes; y los outline permiten formas no rectangulares.

  El propósito de diseño de los *outline* es indicar de forma clara el elemento (botón, enlaces, otros controles…) que está seleccionado en cada momento. Permite establecer el borde de un elemento, definiendo, anchura, color y estilo.

  Los *outline* se definen de forma análoga al borde, definiendo anchura, color y estilo.

  En la figura 2-11, podemos observar el resultado de aplicar distintos estilos de borde a un outline del siguiente ejemplo:

```
p.a{
    color: olive;
     outline-color:#00ff00;
     outline-style:dotted;
}

p.b{
    color: olive;
    outline-color:#00ff00;
    outline-style:ridge;
}

p.c{
    color: olive;
    outline-color:#00ff00;
    outline-style:dashed;
}
```

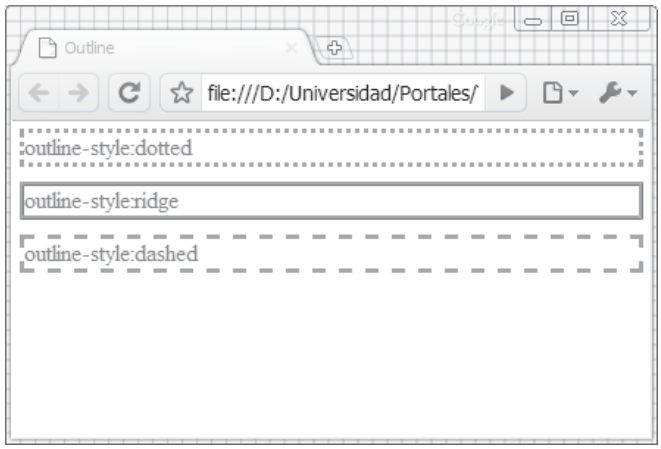


Figura 2-11. Ejemplo de uso del atributo outline-style

2.4.6 Dimensiones en CSS

Las propiedades para establecer dimensiones en CSS nos permiten controlar el tamaño de un elemento en pantalla, por ejemplo: anchura, altura, valor máximo y mínimo de anchura, etc. Para ello CSS implementa los siguientes atributos, que vamos a ver a continuación:

- *width*. Permite establecer la anchura de un elemento. Puede aplicarse a elementos que contienen texto, pero es más útil para elementos reemplazados tales como imágenes. La anchura especificada debe ser respetada, escalando la imagen si es necesario. Cuando se escale, si la propiedad *height* es *auto*, deben preservarse las proporciones de la imagen.
- *max-width*. Permite definir una anchura máxima a la que escalar el elemento en pantalla. Así evitamos que el elemento tome dimensiones no deseadas al maximizar la pantalla.
- *min-width*. La propiedad *min-width* tiene un funcionamiento análogo a la propiedad anterior, sólo que en este caso, la propiedad limita la anchura mínima que debe tener un elemento.
- *height*. Establece la anchura de un elemento. La altura debe ser respetada, debiéndose escalar la imagen si es necesario. Al escalar, si el valor de la propiedad *width* es *auto* se mantienen las proporciones de la imagen.

  Cuando se aplique a un elemento textual, la altura puede ser respetada, por ejemplo mediante una barra de scroll, por ello se debe prestar bastante atención para evitar efectos no deseados.
- *max-height*. Permite definir una altura máxima a la que escalar el elemento en pantalla. Así evitamos que el elemento tome dimensiones no deseadas al maximizar la pantalla.
- *min-width*. Limita la anchura mínima que debe tener un elemento.

2.5 POSICIONAMIENTO

2.5.1 Elementos flotantes

Los elementos flotantes permiten maquetar nuestra Web indicando la posición de los diferentes elementos que forman la web. Para ello, CSS implementa dos atributos:

- *float.* Permite hacer flotar cualquier elemento a la derecha o a la izquierda. Es decir, el elemento se convierte en una caja, según el modelo de cajas, y puede flotar bien a la derecha o a la izquierda, dejando el espacio restante para el resto de elementos.

  Para que la propiedad funcione correctamente, y flote, no podemos olvidar definir la anchura del elemento. En caso contrario permanece como cualquier otro elemento.

  Con el valor *none*, el elemento se muestra donde aparece en el texto. Con el valor *left* el elemento se mueve a la izquierda y el texto continuará por el lado derecho del elemento. Con un valor *right* el elemento se mueve a la derecha y el texto cambiará de línea por el lado izquierdo del elemento.

  Con los ejemplos de la figura 2-12 y 2-13 podemos entender mejor el funcionamiento del elemento *flota*. En la primera figura se han creado dos columnas de texto de 150 píxels de anchura, dejando el espacio restante para demás elementos, en este caso, otro párrafo. En la figura 2-13, se muestra el uso de la propiedad para maquetar un texto alrededor de una imagen, el código se muestra a continuación.

```
<HTML>
<HEAD>
<TITLE>
          Float
</TITLE>
<STYLE>
 img{
     float: left;
        width:250px;
 }
 p{
    clear: left;
}
</STYLE>
</HEAD>
<BODY>
<img src='img.jpg' alt='Imagen de prueba'>
<p>Esto es un texto Esto es un texto Esto es un texto... </p>
</BODY>
</HTML>
```

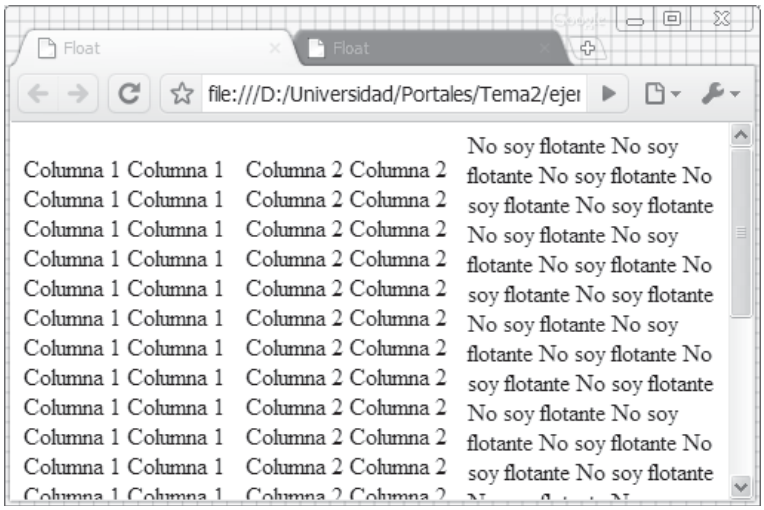


Figura 2-12. Ejemplo de uso del atributo float con texto

Figura 2-13. Ejemplo de uso del atributo float con imágenes

- *clear*. Esta propiedad especifica si un elemento permite que haya elementos flotantes a sus lados. Más concretamente, el valor de esta propiedad enumera los lados en los que no acepta elementos flotantes. Con *clear*:*left* el elemento será movido hasta quedar por debajo del elemento flotante a la izquierda. Con *clear:none* se permiten elementos flotantes a ambos lados.

Si aplicamos el atributo *clear:left* al ejemplo de la figura 2-13, obtendremos como resultado que el texto que se encontraba a la derecha de la imagen pasa a la posición inferior de la imagen.

2.5.2 Visualización

Las propiedades de visualización permiten describir la forma en la que un elemento se muestra en pantalla, definir si es o no es visible, así como definir una estructura por capas de los distintos elementos, obteniendo una tercera dimensión.

A continuación vamos a ver las propiedades más importantes:

- *display.* Describe si un elemento se muestra en el lienzo (ya sea una página impresa, una pantalla de computadora, etc.) y cómo se muestra.

  Un elemento con *display:block* abre un nuevo cuadro. El cuadro se posiciona relativamente a los cuadros adyacentes de acuerdo al modelo de formato de CSS. Normalmente, elementos tales como *<h1>* y *<p>* son del tipo *block*. El valor *list-item* es similar a *block*, excepto que se añade un marcador de objeto de lista. En HTML, *<li>* tiene normalmente este valor.

  Un valor de *none* desactiva la presentación del documento, incluyendo los elementos hijos y el cuadro circundante.

- *visibility*. Permite hacer visibles o invisibles los elementos. Esa propiedad es parecida en funcionamiento a la propiedad *display*, aunque difiere en funcionalidad. En el caso de display, cuando un elemento desaparece, deja su lugar para que sea ocupado por el resto. Sin embargo con *visibility* cuando hace invisible un elemento, el espacio ocupado por el elemento no es ocupado por el resto.

  El valor *collapse* tiene la misma funcionalidad que *hidden* pero sólo se aplica a tablas.

  En el ejemplo de la figura 2-14, podemos comprobar cómo la imagen introducida no es visible, pero sin embargo, se sigue manteniendo el lugar ocupado por ella. El código para conseguir este ejemplo lo podemos ver a continuación.

```
img{
     visibility: hidden;
     width:250px;
}
```

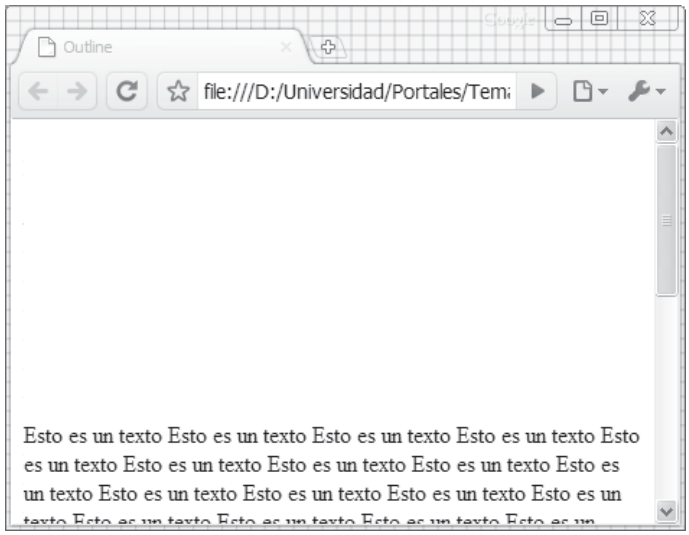


Figura 2-14. Ejemplo de uso del atributo visibility

- *z-index.* Permite crear una tercera dimensión en nuestros diseños, permitiendo apilar capas, es decir, colocar una capa o elemento encima de otro.

  Para establecer la tercera dimensión es necesario que el elemento esté posicionado en el lienzo, además de tener un número de capa que se le puede asignar automáticamente mediante el valor *auto* o manual.

  Con el ejemplo de la figura 2-15, podemos entender mejor su funcionamiento. El código necesario se muestra a continuación.

```
<HTML>
<HEAD>
 <TITLE>
 Z-index
 </TITLE>
<STYLE>
img.capa1 {
    position:absolute;
    top:0px;
    left:0px;
    z-index:1;
}
img.capa2 {
    position:absolute;
    top:50px;
    left:50px;
    z-index:2;
}
img.capa3 {
    position:absolute;
    top:100px;
    left:100px;
    z-index:3;
```

```
}
img.capa4 {
    position:absolute;
    top:150px;
    left:150px;
    z-index:4;
}
P{
    clear: left;
}
</STYLE>
</HEAD>
<BODY>
 <img class='capa1' src='img.jpg' alt='Imagen de prueba'>
 <img class='capa2' src='img.jpg' alt='Imagen de prueba'>
 <img class='capa3' src='img.jpg' alt='Imagen de prueba'>
 <img class='capa4' src='img.jpg' alt='Imagen de prueba'>
 </BODY>
</HTML>
```

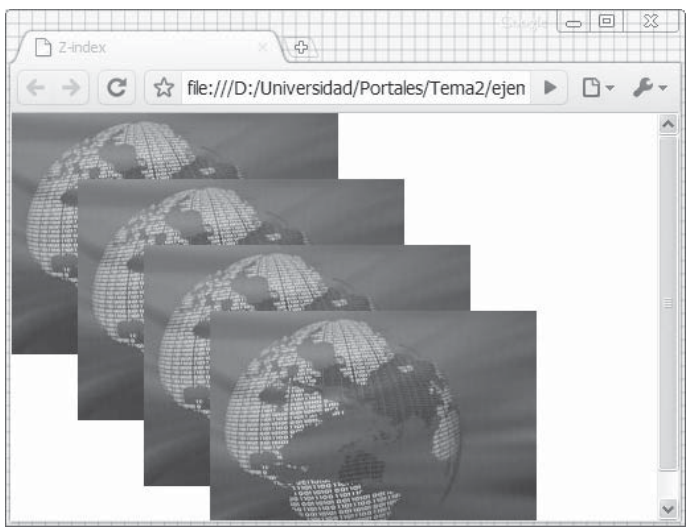


Figura 2-15. Ejemplo de uso del atributo z-index

2.5.3 Esquemas (layouts)

Con todo el conocimiento adquirido a lo largo del capítulo, ya somos capaces de crear los esquemas de distribución de un portal Web, sin la necesidad de utilizar las tablas como base de la maquetación, introduciendo elementos innecesarios y complicando el diseño y el mantenimiento.

Por tanto a lo largo de este apartado, vamos a ver cómo se crea un layout sencillo.

Para saber más. Ejemplos de layout para diseñar tu web

http://layouts.ironmyers.com/

En primer lugar antes de empezar con la maquetación, tenemos que tener en mente el diseño a seguir, para este ejemplo vamos a seguir la distribución de la figura 2-16.

Una vez definido el resultado final del portal, procederemos a crear el código CSS necesario para implementarlo.

Comenzamos definiendo la cabecera:

- La cabecera no debe tener ningún elemento ni a la izquierda ni a la derecha. Por tanto se hace uso del atributo *clear*, para evitar que se pueda colocar algún elemento flotante.
- Establecemos la anchura en píxels de la cabecera.
- Definimos el color.
- Para terminar se establece el color del texto.

```
cabecera{
    clear: both;
    height:80px;
    background:#000000;
    color:#FFFFFF;
}
```

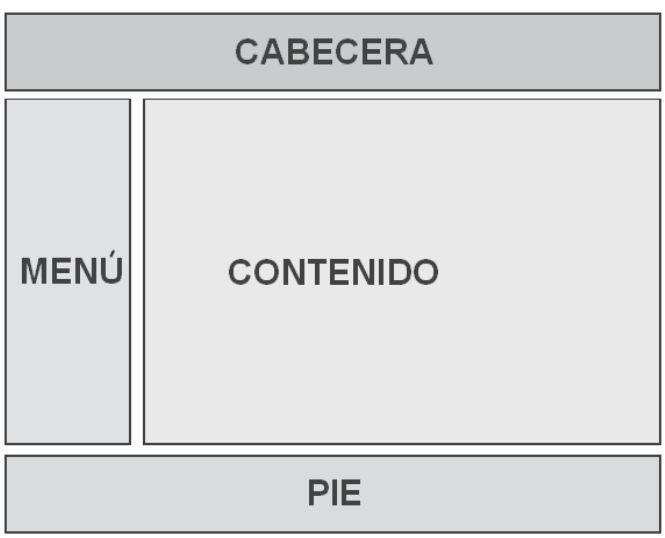


Figura 2-16. Ejemplo del esquema de diseño de un portal Web

Una vez definida la cabecera procedemos a definir el menú:

- Definimos la anchura del mismo, para que ésta siempre quede fija. Sin embargo no se hace lo mismo con la altura, por tanto al no fijarla no limitamos que el menú pueda seguir creciendo.
- Definimos el relleno
- Establecemos el color del borde y su anchura
- Declaramos el elemento como flotante, a la izquierda del lienzo
- Establecemos un color de texto y de fondo

```
menu {
    width: 200px;
    padding: 5px;
    border-color:#999999;
    border-width:8px;
    float: left;
    background:#333333;
    color:#FFFFFF;
}
```

El siguiente paso sería definir el cuadro de contenido, para ello sólo vamos a definir la posición del margen izquierdo.

Para el pie de nuestro portal procederíamos de forma análoga a la definición de la cabecera, cambiando colores.

Una vez definidos los distintos elementos del portal, vamos a crear un nuevo elemento (contenedor) o cuadro que los contenga a todos y sobre el que asignar los atributos comunes. Para el *id* contenedor se han definido los siguientes atributos:

- Anchura del contenedor
- Márgenes
- Alineación del texto
- Color de fondo

```
contenedor {
    width: 700px;
    margin: 10px 10px;
    text-align: left;
    background:#CCCCCC;
}
```

Una vez definidos todos los elementos, ya estamos en disposición de generar nuestra primera página del portal tal y como podemos ver en la figura 2-17 y cuyo código se muestra tras la figura.

Como podemos comprobar, acabamos de diseñar nuestro primer portal, haciendo la distribución del mismo exclusivamente con las hojas de estilo, y por tanto cumpliendo el estándar del W3C.

Cuando queramos conocer si nuestro diseño de hojas de estilo cumple o no con el estándar de la W3C, podemos hacer uso de los validadores que suministra para comprobar la compatibilidad.

Para saber más. Validador CSS del W3C

http://jigsaw.w3.org/css-validator/

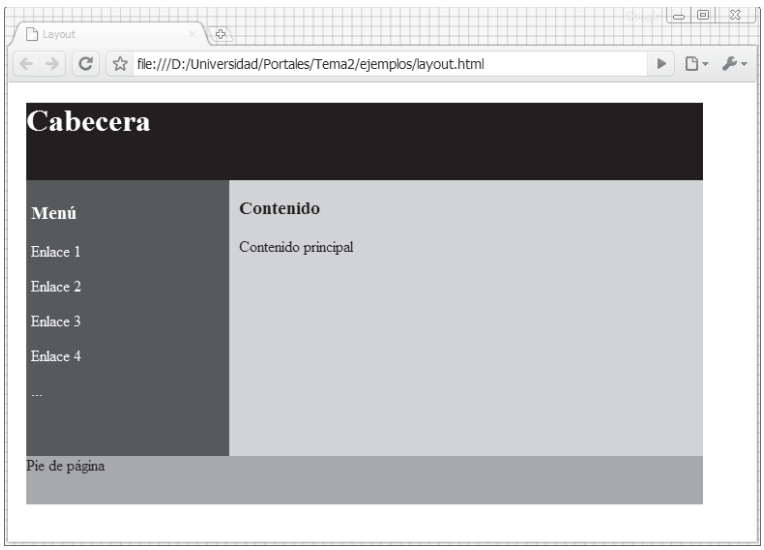


Figura 2-17. Primera Web del portal

```
<BODY>
<div id="contenedor">
    <div id="cabecera">
        <h1>Cabecera</h1>
    </div>
    <div id="menu">
        <h3>Men&uacute;</h3>
        <p>Enlace 1</p>
        <p>Enlace 2</p>
        <p>Enlace 3</p>
        <p>Enlace 4 </p>
        <p>...</p>
        <p> </p>
    </div>
    <div id="contenido">
        <h3>Contenido</h3>
        <p>Contenido principal</p>
    </div>
    <div id="pie">
        <p>Pie de página</p>
    </div>
</div>
</BODY>
```

2.6 RESUMEN

A modo de resumen, en la tabla 2-6 se pueden ver las propiedades más importantes de las hojas de estilo.

Tabla 2-6. Resumen CSS

Propiedad	
Propiedades de fuente	
Font-family	Font-weight
Font-style	Font-size
Font-variant	Font
Propiedades de color y fondo	
Color	Background-attachment
Background-color	Background-position
Background-image	Background
Background-repeat	
Propiedades de texto	
Word-spacing	Text-transform
Letter-spacing	Text-align
Text-decoration	Text-indent
Vertical-align	Line-height
Propiedades de cuadro	
Margin-top	Border-left-width
Margin-right	Border-width
Margin-bottom	Border-color
Margin-left	Border-style
Margin	Border-top
Padding-top	Border-right
Padding-right	Border-bottom
Padding-bottom	Border-left
Padding-left	Border
Padding	Width
Border-top-width	Height
Border-right-width	Float
Border-bottom-width	Clear
Propiedades de clasificación	
Display	List-style-position
White-space	List-style
List-style-type	

2.7 UTILIZACIÓN DE HOJAS DE ESTILO CON ECLIPSE

Para crear páginas web y hojas de estilo podemos utilizar cualquier editor de textos aunque se recomienda utilizar herramientas que nos faciliten el trabajo. En este apartado vamos a ver el uso de una herramienta Eclipse que ayuda a mantener, organizar y crear un proyecto Web estático.

Las ventajas de este tipo de herramientas frente al uso de un editor de textos normal son bastante notables, ya que ayudan y facilitan el desarrollo de los proyectos, gracias a pequeños detalles como generación de código, formateado de texto, comprobación de errores, etc.

2.7.1 Instalación

En primer lugar vamos a instalar el entorno de programación Eclipse en la distribución *Fedora* haciendo uso del comando *yum*.

```
yum install eclipse
```

Una vez instalado el entorno de programación, debemos instalar el plugin para poder trabajar con páginas Web. Para ello debemos ir al menú de *Ayuda* y pulsar sobre *Install New Software.*

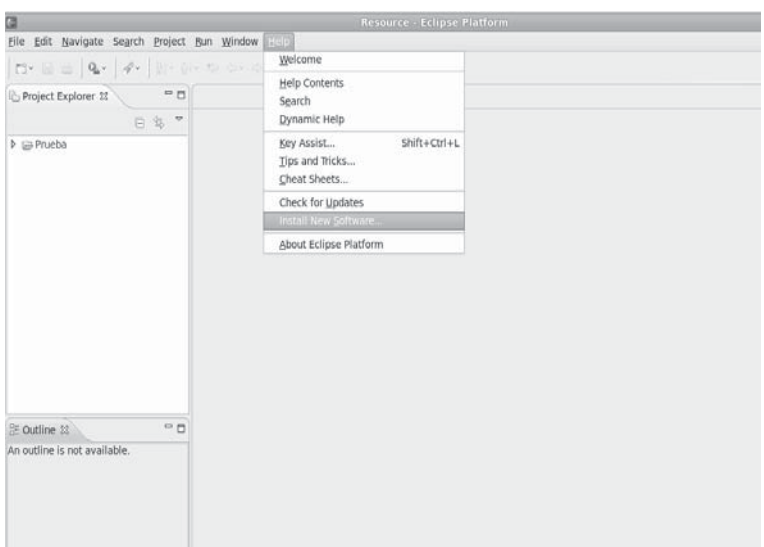


Figura 2-18. Instalando plugins en Eclipse

En el campo *work*-with introducimos la URL oficial del plugin de Eclipse (http://download.eclipse.org/releases/galileo) para poder acceder al repositorio y buscar el plugin adecuado para nuestro lenguaje de programación. En este caso se busca y

selecciona el plugin: **Eclipse Web Developer Tools**, plugin válido para generar documentos HTML, XML, CSS, JavaScripts, etc.

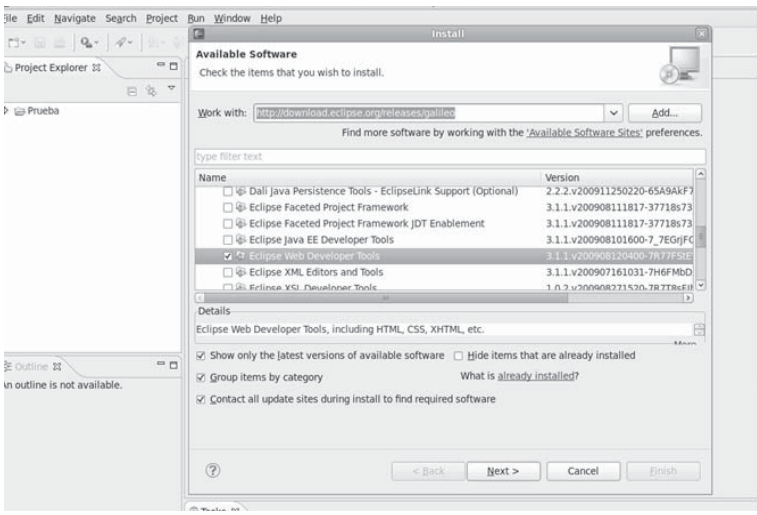


Figura 2-19. Selección del plugin

Para proceder con la instalación pulsamos el botón *Next*, aceptamos las condiciones de la licencia y se finaliza la instalación (véase la figura 2-20).

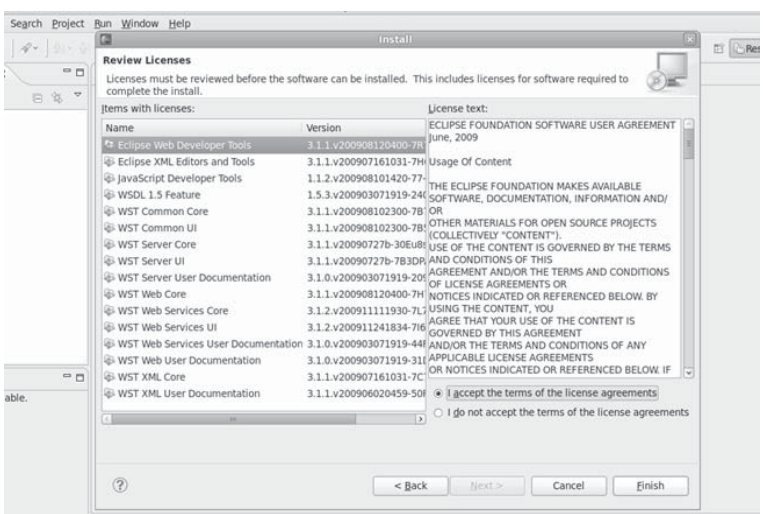


Figura 2-20. Terminando con la instalación del plugin

2.7.2 Utilización del Eclipse Web Developer Tools

Para empezar a trabajar con Eclipse, el primer paso es crear el proyecto de trabajo. Para ello, pulsamos sobre el menú *File>New>Project*, para abrir el asistente de selección de proyectos y seleccionamos *Static Web Proyect*, abriéndose el asistente de configuración del proyecto (véase la figura 2-21).

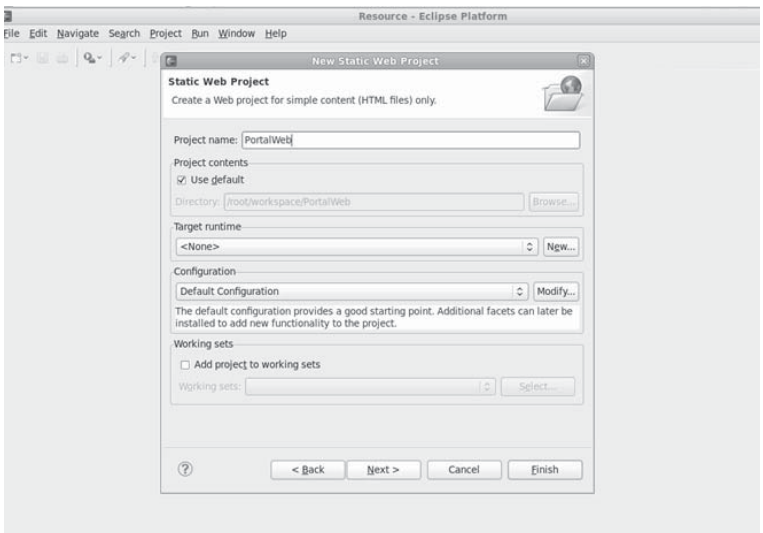


Figura 2-21. Creación de un nuevo proyecto de páginas web estáticas

Le asignamos un nombre al proyecto y opcionalmente un directorio de trabajo, en caso de no asignarlo, crea un directorio con el nombre del proyecto en el directorio de trabajo de Eclipse.

En este momento ya hemos creado el proyecto, el cual se puede ver en la pestaña *Proyect explorer* de Eclipse, si pulsamos sobre el nombre del proyecto, éste se despliega y se pueden ver en forma de árbol los distintos elementos que lo componen: contenidos, librerías, etc.

Para trabajar con CSS, basta con añadir un nuevo archivo al proyecto de tipo CSS. Para ello pulsamos el botón derecho sobre la carpeta *WebContent, New* y *Other* y se muestra la ventana de selección que se muestra en la figura 2-22 y donde seleccionamos el tipo CSS.

Una vez seleccionado el tipo de archivo se inicia un asistente, en este caso para CSS, donde debemos configurar los parámetros como: ruta de almacenamiento, nombre del fichero, codificación, plantilla por defecto, etc. En la figura 2-23, podemos ver el último paso del asistente donde se seleccionan las plantillas a seguir para la creación del fichero. En nuestro caso utilizaremos la plantilla del nuevo archivo CSS.

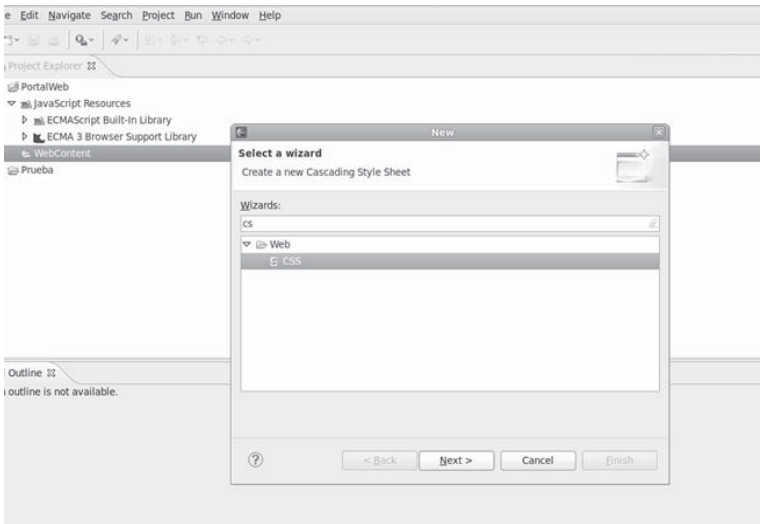


Figura 2-22. Nuevo archivo CSS

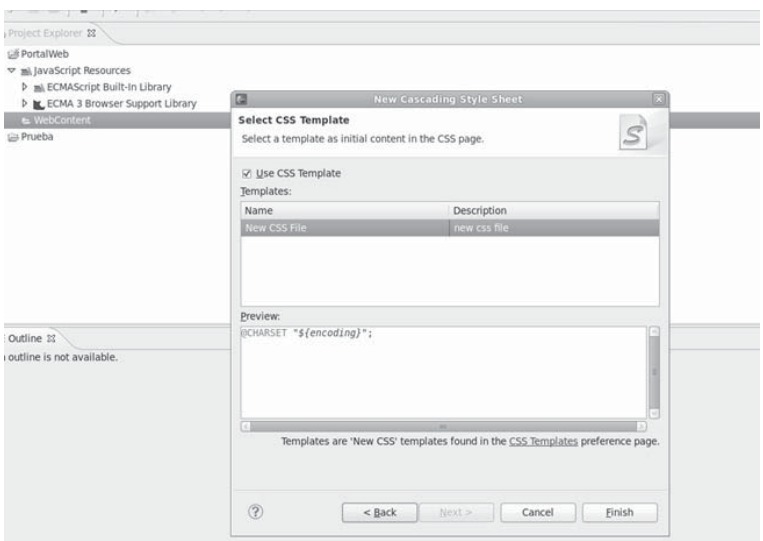


Figura 2-23. Configurador CSS

Tras crear el archivo, éste aparece en la estructura de árbol del proyecto en la carpeta *WebContent*, o en cualquier otra si se cambió la ruta por defecto. Haciendo doble clic, lo abrimos y se puede comenzar a editarlo, en la figura 2-24 podemos ver un ejemplo de cómo el plugin formatea el código para hacerlo más legible.

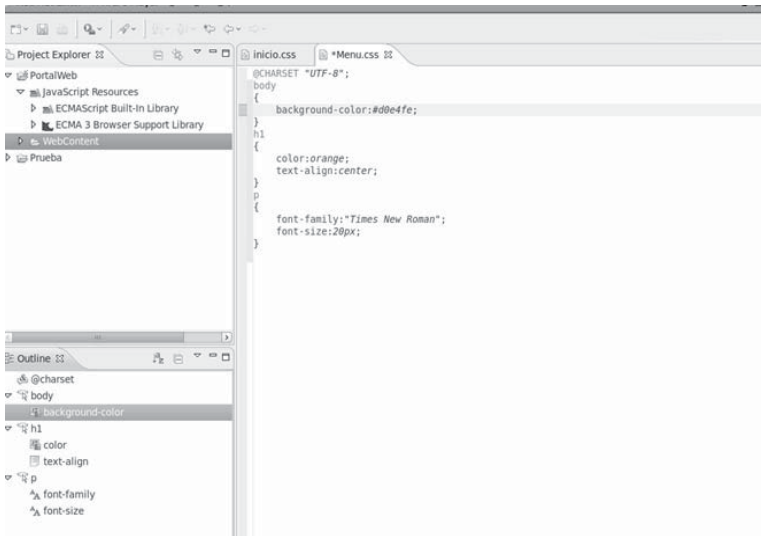


Figura 2-24. Archivo CSS en Eclipse

En la figura 2-25 podemos ver otra de las funcionalidades que nos aporta el plugin, que nos permite autocompletar el código.

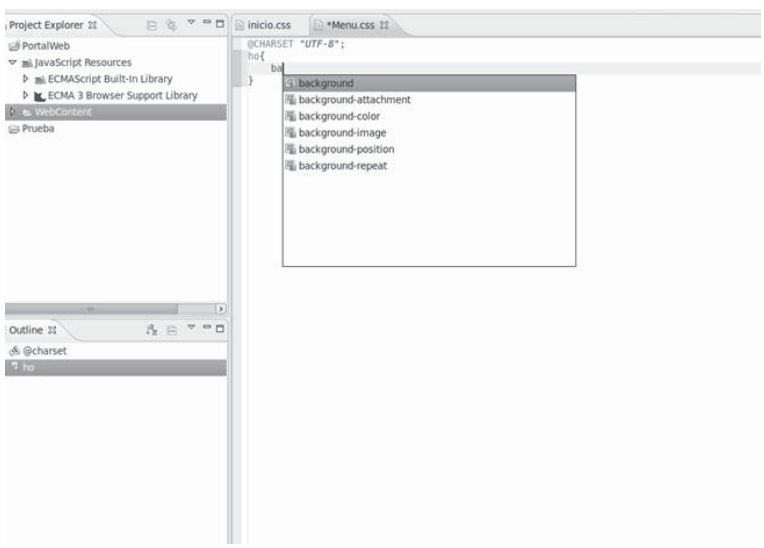


Figura 2-25. Autocompletar código con Eclipse Web Developer Tools

Capítulo 3

SERVIDORES DE APLICACIONES LAMP

3.1 INTRODUCCIÓN

Hoy en día los servidores de aplicaciones son identificados fundamentalmente con gestores de contenidos web, sistemas de comercio electrónico o sistemas para la construcción de sitios web. Los servidores LAMP presentan dos características que permiten diferenciarlos de forma categórica de otros servidores (en especial servidores web), como son su fuerte integración con bases de datos y el gran dinamismo de los contenidos y servicios que ofrecen.

Una de las grandes ventajas de los servidores de aplicaciones es la facilidad para implantarlos y desarrollarlos, pues por lo general se componen de módulos o herramientas que no necesitan ser programadas. Por ejemplo, una wiki es un ejemplo de servidor de aplicaciones en el que sus usuarios pueden editar el contenido de la misma a partir de los artículos.

Un servidor LAMP es un caso particular de servidores de aplicaciones. Con el acrónimo LAMP hacemos referencia a un conjunto de diferentes herramientas que utilizadas de manera conjunta nos permiten implantar de una forma rápida y eficiente soluciones de servidor de páginas web dinámicas. Estas herramientas son:

- **Linux (L).** Servidor GNU/Linux donde se instalan los servicios.
- **Apache (A).** Servidor de páginas web.
- **MySQL (M).** Servidor de base de datos para permitir que las páginas web sean dinámicas.
- **PHP (P).** Lenguaje de programación usado para la creación de la web dinámica.

En ocasiones alguna de las herramientas es sustituida por otra dando lugar a otros posibles acrónimos para referirse a este mismo tipo de servidores, como por ejemplo

WAMP (con un servidor de la familia Microsoft Windows en lugar de Linux), o incluso LAMA (con Linux, Apache, MySQL, y ASP).

Figura 3-1. Logo de los sistemas LAMP

Los servidores LAMP son muy populares gracias a que las diferentes herramientas que integran la solución global son software libre, proporcionando una plataforma de desarrollo de bajo coste con una gran funcionalidad y capacidad de adecuación a nuestras necesidades. Su uso es por tanto bastante común para la puesta en marcha de portales web en empresas, ya que permiten de una forma segura y transparente el acceso a sus servicios web de forma centralizada, prácticamente desde cualquier dispositivo.

3.2 PASOS PREVIOS

Asumimos de aquí en adelante que disponemos de un servidor Linux con la distribución Fedora Core 13 instalada en él (L de Linux). Para poner en marcha un servidor de aplicaciones LAMP debemos instalar y por este orden MySQL, el servidor web Apache y finalmente PHP.

Antes de iniciar el proceso de instalación de cada uno de los componentes del servidor LAMP es recomendable que actualices antes el sistema ejecutando:

```
yum update
```

3.3 APACHE

Apache (http://www.apache.org/) es el componente servidor web de nuestro servidor de aplicaciones LAMP. De sobra son conocidas las enormes posibilidades que ofrece y las características que presenta este servidor web, de código abierto y multiplataforma (puede ser usado en sistemas UNIX, Windows, Macintosh, etc.).

El servidor Apache es desarrollado dentro del proyecto HTTP Server (httpd) de la Apache Software Foundation, siendo el más utilizado en servidores web desde el año 1996. Una de las principales características del servidor Apache es su arquitectura modular, lo que permite adaptar la funcionalidad del mismo en función de nuestras necesidades fácilmente. El servidor consta, por tanto, de una sección core y diversos módulos que aportan su funcionalidad.

3.3.1 Instalación

Para instalar el servidor *Apache2* fácilmente desde repositorios podemos ejecutar el siguiente comando:

```
# yum install httpd
```

Indicamos también que sea iniciado en el arranque del sistema operativo:

```
# chkconfig --levels 235 httpd on
```

Iniciamos el servidor ejecutando:

```
# service httpd start
```

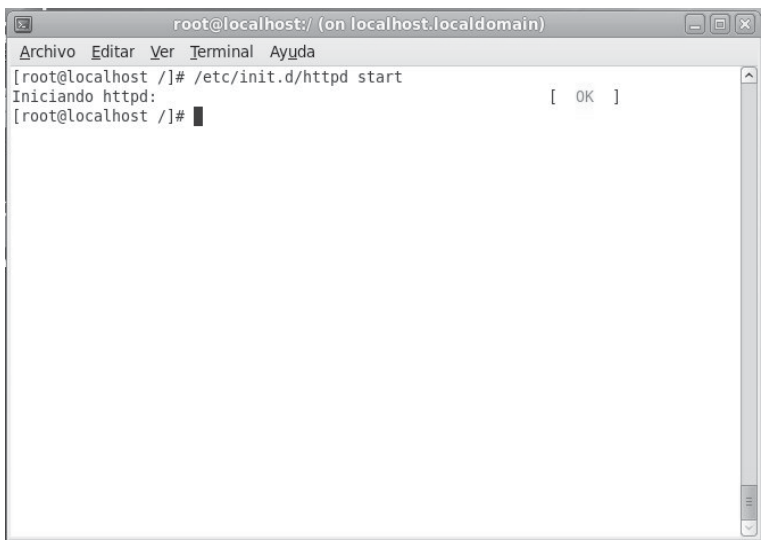


Figura 3-2. Apache – Inicio del servicio

3.3.2 Configuración

La raíz principal para los documentos del servidor web Apache está ubicada en el directorio */var/www/html* en Fedora. La configuración del mismo se realiza editando el fichero de configuración */etc/httpd/conf/httpd.conf*. Por ejemplo, las primeras tareas se encuentran enfocadas a la configuración de las siguientes *características:*

- Para configurar la dirección IP y el nombre del servidor, se deben añadir las siguientes directivas:

```
Listen 127.0.0.1:80
ServerName www.ejemplo.es
```

- *DocumentRoot*. Indica al proceso servidor el directorio en el que se sitúan las páginas HTML a servir. Por defecto, la directiva presenta el valor:

```
DocumentRoot     /var/www/html
```

- *ServerRoot*. Indica al proceso servidor el directorio en el que se encuentran los ficheros necesarios para la configuración del servidor:

```
ServerRoot     /etc/httpd
```

- *DirectoryIndex*. Establece la página (o páginas) predeterminada que se enviará al cliente cuando éste solicita el índice de un directorio:

```
DirectoryIndex     index.html index.phtml index.htm
```

- *User/Group*. Establece el usuario y grupo al que pertenecen los procesos que ejecute el servidor:

```
User apache
Group apache
```

- *LoadModule*. Permite incrementar la funcionalidad del servidor mediante la inserción de librerías:

```
LoadModule     auth_module modules/mod_auth.so
```

- *ErrorDocument*. Modifica la respuesta del servidor ante un error de HTTP:

```
ErrorDocument  500      "Error del servidor. Avise al administrador"
ErrorDocument  403      /etc/httpd/errores/noencontrado.html
```

Algunas de estas directivas se pueden configurar con la herramienta *system-config-httpd*. Para acceder a la web principal de nuestro servidor escribimos en la barra de direcciones http://localhost/ o http://dirección_ip/.

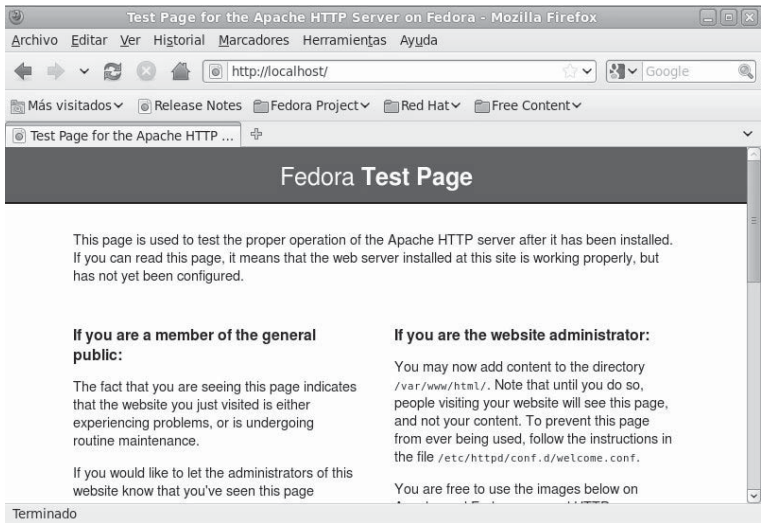


Figura 3-3. Apache – Web inicial de prueba

3.3.3 Permisos

A continuación vamos a ver las dos formas de limitar el acceso a un directorio en Apache: por usuarios o por dirección IP.

3.3.3.1 RESTRINGIR EL ACCESO DE USUARIOS

Es posible restringir el acceso a un determinado directorio de nuestro sitio web Apache para un determinado grupo de usuarios. En nuestro caso el directorio que deseamos proteger es */var/www/html/myadmin*.

Para comenzar debemos crear un fichero con el nombre .htaccess (con un punto delante, para que permanezca oculto) en el directorio a proteger. El contenido de este fichero debe ser el siguiente:

```
AuthUserFile /var/www/html/myadmin/.htpasswd
AuthGroupFile /dev/null
AuthName "Directorio Protegido"
AuthType Basic

Options All
<Limit GET POST PUT>
require valid-user
</Limit>
```

Como podemos ver en el código, la autenticación especificada es básica. El fichero al que hacemos referencia mediante *AuthUserFile* es el que contiene los usuarios y

contraseñas con acceso al directorio. Para generar este fichero usamos la utilidad *htpasswd*, que nos permite generar un fichero de claves,

```
# htpasswd -c /var/www/html/myadmin/.htpasswd usuario
New password:
Re-type new password:
Adding password for user usuario
```

donde la opción –c permite crear el fichero de contraseñas en blanco y *usuario* es el nombre de usuario que queremos crear.

Para restringir finalmente el acceso al directorio en Apache creamos la siguiente directiva en el fichero de configuración */etc/httpd/conf/httpd.conf.*

```
<Directory "/var/www/html/myadmin">
    AllowOverride All
</Directory>
```

Terminamos el proceso reiniciando el servidor Apache:

```
# service httpd restart
```

3.3.3.2 RESTRINGIR EL ACCESO DE REDES Y DIRECCIONES IP

Si lo que queremos es restringir el acceso a un recurso (directorio de nuestro sitio web) a una determinada dirección de red o de equipo, esto también lo configuramos de forma similar en el fichero de configuración de Apache *httpd.conf*. Por ejemplo si deseamos restringir el acceso al directorio *protegido* de nuestra Intranet para permitir solamente la red 176.16.0.0/16, incluimos la siguiente directiva:

```
<Directory /var/www/html/protegido>
   Order Deny,Allow
   Deny from all
   Allow from 176.16.0.0/16
</Directory>
```

Para hacerlo a una determinada dirección de *host* lo hacemos de forma análoga, escribiendo la dirección del *host* en lugar de la dirección de red:

```
<Directory /var/www/html/protegido>
   Order Deny,Allow
   Deny from all
   Allow from 127.0.0.1
</Directory>
```

Finalmente, reiniciando el servidor Apache:

```
# service httpd restart
```

3.4 MYSQL

MySQL es un sistema de gestión de base de datos relacional, multihilo y multiusuario, desarrollado completamente en C y que se distribuye bajo licencia GNU GPL. Es utilizado en la mayoría de las aplicaciones web, con un uso muy ligado al lenguaje PHP.

Entre sus ventajas podemos destacar:

- Multihilo.
- Multiusuario.
- Libre para aplicaciones no comerciales.
- Soporta un amplio subconjunto del lenguaje SQL.
- Disponible para multitud de plataformas y sistemas.
- Conectividad segura.
- Es muy rápido.
- Es fácil de usar.
- Está muy bien documentado.

Más información

Página web oficial de MySQL
http://www.mysql.com

3.4.1 Instalación

Para instalar MySQL se va a necesitar instalar dos paquetes *mysql-server* y *mysql*. Para ello, basta con ejecutar:

```
# yum install mysql-server mysql
```

Una vez realizada la instalación iniciamos el servidor de base de datos *mysqld* ejecutando:

```
# service mysqld start
```

Si deseamos que el servicio se ejecute automáticamente al arrancar el sistema deberemos ejecutar el comando:

```
# chkconfig mysqld on
```

Nota

Es recomendable que como primer paso establezcamos la contraseña para el usuario *root* con acceso a MySQL.

```
# mysqladmin -u root password 'contrasena'
```

3.4.2 Tareas básicas

Como veremos más adelante, una forma muy cómoda de administrar el servidor de bases de datos es utilizando *PHPMyAdmin* aunque resulta necesario saber realizar las tareas más básicas con el shell del sistema.

3.4.2.1 CAMBIAR LA CONTRASEÑA DEL ADMINISTRADOR

Cuando se instala MySQL el servidor por defecto utiliza el usuario *root* que no tiene contraseña. Lógicamente, es muy recomendable cambiar la contraseña ejecutando el siguiente comando:

```
# mysqladmin –u root password ‘hola00’
```

A partir de ahora para acceder a la shell de MySQL debemos añadir la opción -p.

```
# mysql –u root -P
```

3.4.2.2 COPIAS DE SEGURIDAD DE UNA BASE DE DATOS

Para realizar las copias de seguridad en MySQL se utiliza el comando *mysqldum* que puede utilizarse para volcar una base de datos o colección de bases de datos para copia de seguridad o para transferir datos a otro servidor SQL (no necesariamente un servidor MySQL). EL volcado contiene comandos SQL para crear la tabla y/o rellenarla.

Para realizar una copia de seguridad de una base de datos ejecutamos:

```
# mysqldump -u root -p nombre_base_de_datos > copia_seguridad.sql
```

Una vez realizada la copia de seguridad, si queremos restaurarla podremos ejecutar:

```
# mysql –u root –p < copia_seguridad.sql
```

3.4.3 La shell de MySQL

La shell de MySQL permite interactuar de forma directa con el motor de base de datos. A continuación se va a ver de forma breve el uso de la shell de MySQL, para realizar las operaciones básicas con la base de datos. Aunque a primera vista, resulta un poco difícil, cuando se trabaja con ella se trabaja más rápido. En el siguiente apartado se verán distintos entornos gráficos para trabajar con MySQL.

Antes de empezar a trabajar, el primer paso es conectar con el servidor, para ello se utiliza el siguiente comando:

```
# mysql –h localhost –u root
```

donde –h permite indicar la dirección IP o nombre del servidor y –u permite indicar el nombre de usuario con el que se accede al sistema.

Si se conecta al equipo local entonces debemos ejecutar:

```
# mysql –u root
```

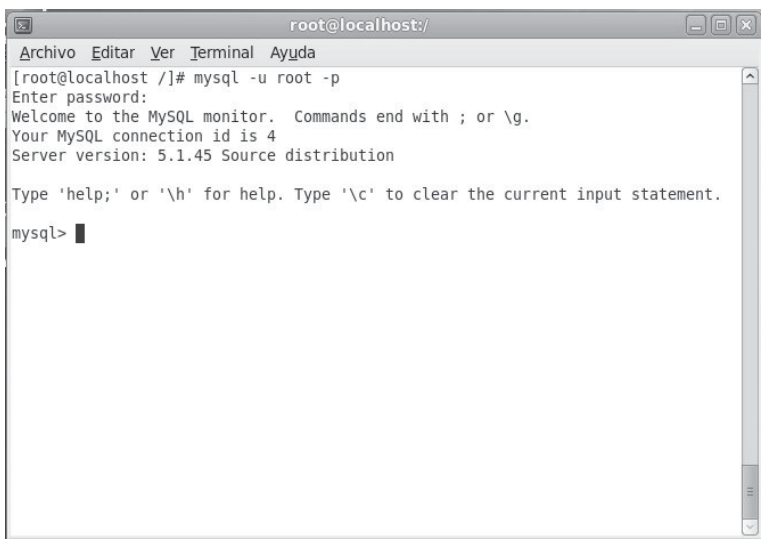


Figura 3-4. Shell de MySQL

A continuación se van a ver las tareas más comunes que debe realizar un administrador de la base de datos.

3.4.3.1 CREACIÓN DE UNA BASE DE DATOS

Para crear una base de datos se usa el comando *CREATE DATABASE nombre.* A continuación en el ejemplo se crea una base de datos llamada agenda.

```
mysql> CREATE DATABASE agenda;
```

Para comprobar que se ha creado correctamente podemos hacer uso del comando *SHOW databases*, que muestra las bases de datos creadas en el servidor.

```
mysql> SHOW databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| agenda             |
| mysql              |
| test               |
+--------------------+
4 rows in set (0.00 sec)
```

3.4.3.2 PERMISOS DE ACCESO

Una vez creada la base de datos se deben definir los permisos de acceso a la base de datos, para ello se utiliza el comando GRANT, cuya sintaxis es la siguiente:

```
mysql> GRANT privilegios
       ON elemento
       TO nombre_usuario
       IDENTIFIED BY 'contrasena'
```

donde:

- **Privilegios**. Especifica las operaciones que puede realizar el usuario (*insert, select, update, delete, create, drop, alter* y *ALL*).
- **Elemento**. Indica la base de datos o tablas sobre las que se aplicarán los permisos.
- **NombreUsuario.** Indica el usuario al que se le da acceso, si no existe lo crea.

En el siguiente ejemplo se asignan todos los permisos al usuario *usuario* para la base de datos *agenda*.

```
mysql> GRANT ALL PRIVILEGES ON agenda TO usuario@localhost
IDENTIFIED BY 'pass';
```

Si por ejemplo, deseamos establecer los permisos al mismo usuario para que tenga acceso desde el equipo 192.168.20.5 entonces debemos ejecutar:

```
mysql> GRANT ALL PRIVILEGES ON agenda TO usuario@192.168.20.5
IDENTIFIED BY 'pass';
```

Nota

Los permisos de acceso en MySQL van establecidos por la tupla (equipo/usuario/contraseña).

3.4.3.3 CREAR TABLAS

Para crear una tabla dentro de una base de datos antes debemos seleccionarla con el comando use. A continuación vamos a indicar que vamos a empezar a trabajar con la base de datos *agenda*:

```
mysql> USE agenda;
```

Una vez seleccionada la base de datos, ya podemos empezar a trabajar con ella. En primer lugar, vamos a crear una tabla llamada *agenda*, con los siguientes campos: *teléfono, nombre, apellidos, dirección y email*. A continuación se muestra el código de ejemplo para crear la tabla agenda.

```
mysql> CREATE TABLE agenda
 ->(telefono VARCHAR(12), nombre VARCHAR(20),
 -> apellido VARCHAR(80), direccion VARCHAR(100),
 ->email VARCHAR(50));
   Query OK, 0 rows affected (0.07 sec)
```

Tras crear la tabla podemos comprobar que ha sido creada usando el comando *SHOW TABLES*, que muestra un listado de todas las tablas de la base de datos. A continuación se puede ver un ejemplo, y se puede ver cómo aparece la tabla que se acaba de crear.

```
mysql> SHOW TABLES;
+------------------+
| Tables in agenda |
+------------------+
| agenda           |
+------------------+
1 row in set (0.00 sec)
```

3.4.3.4 CONSULTAS SQL

Tras crear la base de datos y la tabla ya estamos en disposición de insertar datos en la tabla, para ello se van a insertar un par de registros. A continuación se muestra un ejemplo de inserción.

```
mysql> INSERT INTO agenda
    -> (telefono, nombre, apellido, direccion, email)
    -> VALUES ('12356789', 'José', 'García García',
 -> 'Calle Nueva nº 23','jo@yahoo.es');
   Query OK, 1 row affected (0.04 sec)
```

Una vez introducidos los datos se puede realizar algún ejemplo de selección de datos, viendo cómo nos lo devuelve. A continuación vamos a crear un ejemplo que muestra todos los datos de la tabla agenda.

```
mysql> select * from agenda;
+-----------+---------+-----------------+----------------------+-----
-------------+
| telefono  | nombre  | apellido        | direccion            |
email            |
```

```
+-----------+---------+-----------------+---------------------+-----
-------------+
| 12356789  | José    | García García   | Calle Nueva nº 23   |
jo@yahoo.es      |
| 324445559 | Antonio | Moreno García   | Calle prueba nº 23 |
amoreno@yahoo.es |
+-----------+---------+-----------------+---------------------+-----
-------------+
2 rows in set (0.00 sec)
```

Como se ha podido comprobar, la shell de MySQL es una herramienta bastante útil para mantener una base de datos, así como para verificar las consultas SQL, antes de introducirlas en programas, pudiendo así asegurar su correcto funcionamiento.

Si deseas más información sobre el lenguaje SQL puedes consultar el apéndice I.

3.4.4 Interfaces gráficas

En este capítulo describimos de manera breve tres de las interfaces gráficas de usuario más importantes para el desarrollo y mantenimiento de bases de datos MySQL. Este tipo de herramientas son muy importantes ya que facilitan la manera en la que trabajamos con este tipo de bases de datos, mejorando el rendimiento del administrador al aumentar la simplicidad y productividad de sus actividades.

3.4.4.1 PHPMYADMIN

PhpMyAdmin (http://www.phpmyadmin.net/) es una interfaz web para la gestión de bases de datos MySQL. Su uso se presenta fundamental en un servidor de aplicaciones LAMP, ya que permite que la interacción entre usuario y base de datos MySQL sea extremadamente sencilla a la vez que eficiente. Para instalar phpMyAdmin lo hacemos desde repositorios usando *yum*:

```
# yum install phpmyadmin
```

Una vez instalado debemos configurarlo al menos de forma básica para poder operar con él, sobre todo si accedemos a *phpMyAdmin* no sólo desde el equipo local en el que se encuentra instalado. Para permitir el acceso desde otros equipos editamos el fichero de configuración

```
# vi /etc/httpd/conf.d/phpMyAdmin.conf
```

y en la etiqueta *allow from* indicamos la IP o red que queremos que tenga acceso. Si deseamos que tengan acceso todos los equipos entonces pondremos *allow from all*.

```
<Directory /usr/share/phpMyAdmin/>
   order deny,allow
   deny from all
   allow from 192.168.0.0/24 # permite la red interna
</Directory>
```

A continuación reiniciamos el servidor web Apache para que los cambios realizados hasta el momento tengan efecto:

```
# service httpd restart
```

Ahora podemos acceder a phpMyAdmin escribiendo en la barra de direcciones del navegador http://localhost/phpmyadmin/ (véase la figura 3-5).

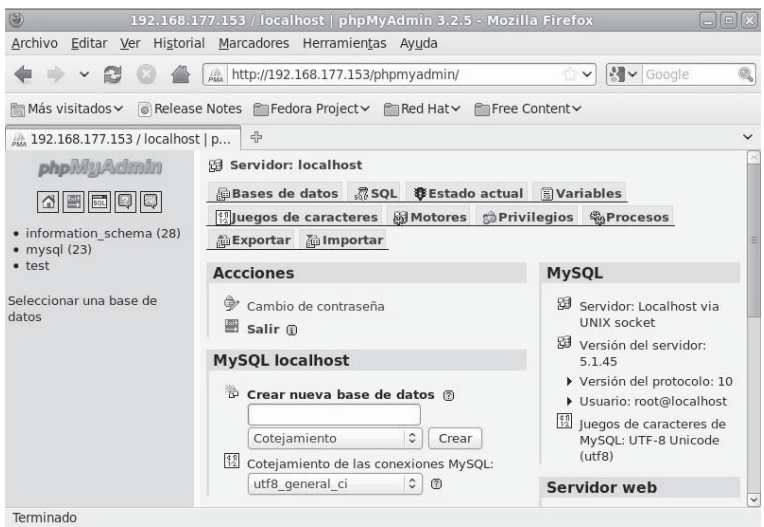


Figura 3-5. phpMyAdmin – Ventana principal de administración

Desde aquí podemos manipular completamente el contenido de nuestras bases de datos MySQL como si estuviéramos manejando la propia consola del gestor. Así, podemos crear nuevas bases de datos, ejecutar código SQL, ver el estado actual de las bases de datos, establecer privilegios de acceso para los diferentes usuarios, etc.

3.4.4.2 MYSQL WORKBENCH

MySQL Workbench (http://wb.mysql.com/) es una herramienta oficial desarrollada por el propio equipo MySQL, bajo la licencia GPL, que ha sustituido recientemente a anteriores herramientas gráficas de MySQL como MySQL GUI Tools Bundle.

MySQL Workbench permite diseñar y modelar bases de datos, desarrollar código SQL, administrar nuestra base de datos MySQL, etc., todo ello gracias a una eficiente interfaz gráfica, muy fácil e intuitiva.

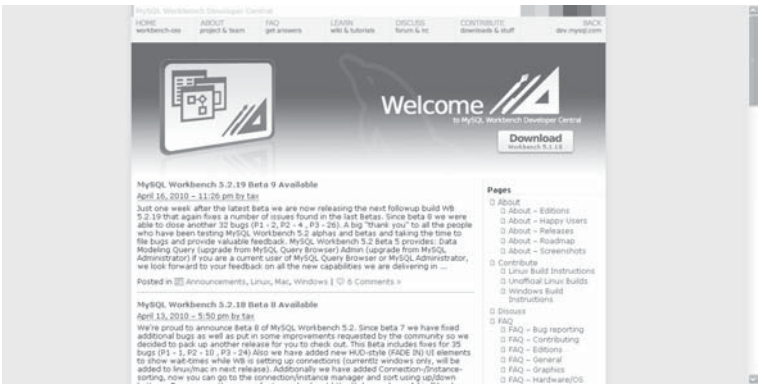


Figura 3-6. Web oficial de MySQL Workbench (http://wb.mysql.com/)

A continuación vamos a ver cómo instalar esta herramienta administrativa imprescindible para bases de datos MySQL tanto en Windows como en Linux (Fedora). Después haremos un breve repaso al entorno visual y las funcionalidades más destacadas de las que dispone.

3.4.4.2.1 Instalación en sistemas operativos Windows

La instalación de esta herramienta en sistemas operativos Microsoft Windows es muy sencilla. Para ello hay que realizar los siguientes pasos:

- Descargamos el ejecutable de la web y hacemos doble clic para comenzar el proceso de instalación.
- Seleccionamos el tipo de instalación (completa o por componentes) y pulsamos *Next.*
- A continuación, seleccionamos la unidad donde deseamos realizar la instalación.
- El sistema nos muestra un resumen del proceso de instalación y pulsamos *Install* para iniciar la instalación (véase la figura 3-7).

Una vez concluida la instalación se nos muestra una ventana final en la que observamos que se ha realizado correctamente.

3.4.4.2.2 Instalación en Fedora Core 13

Para realizar la instalación en Fedora primero tenemos que descargarnos el fichero *rpm* de MySQL Workbench de la página oficial. Pulsamos dos veces sobre el fichero descargado y el gestor de paquetes resolverá las dependencias del paquete y tal como se muestra en la figura 3-8, nos mostrará los paquetes que deben ser instalados. Pulsamos *Continuar* y se iniciará el proceso de instalación.

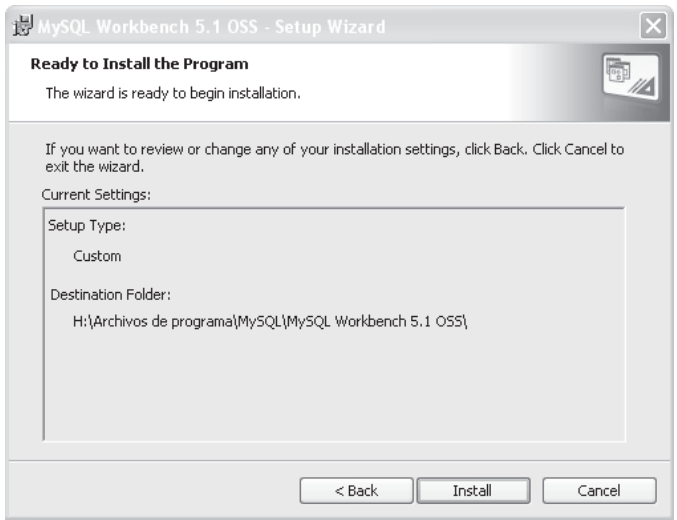


Figura 3-7. MySQL Workbench – Resumen de la instalación

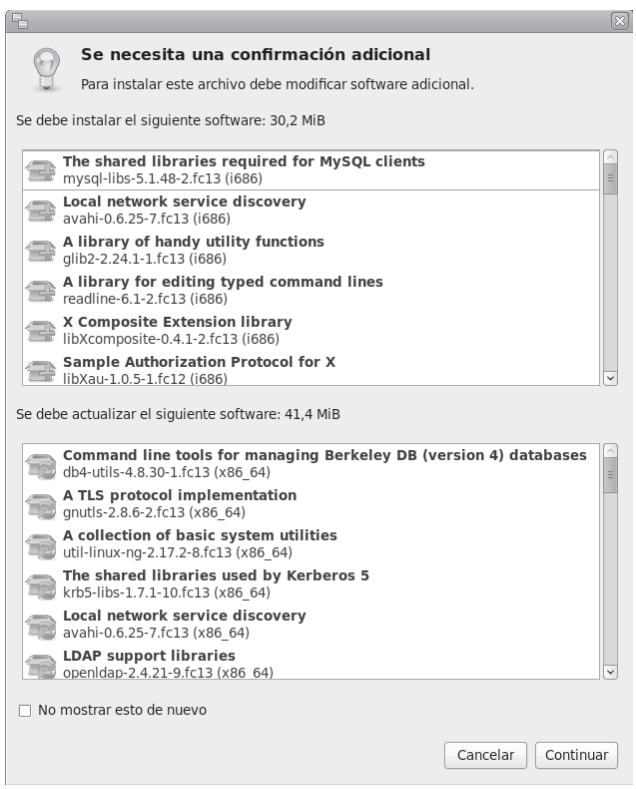


Figura 3-8. MySQL Workbench – Instalación en Fedora

Para iniciar MySQL Workbench podemos hacerlo desde el submenú *Programación* o ejecutando desde el terminal:

```
$ mysql-workbench
```

3.4.4.2.3 Entorno gráfico y funcionalidades principales

En la figura 3-9 podemos apreciar de manera general el entorno gráfico presentado por MySQL Workbench en Windows.

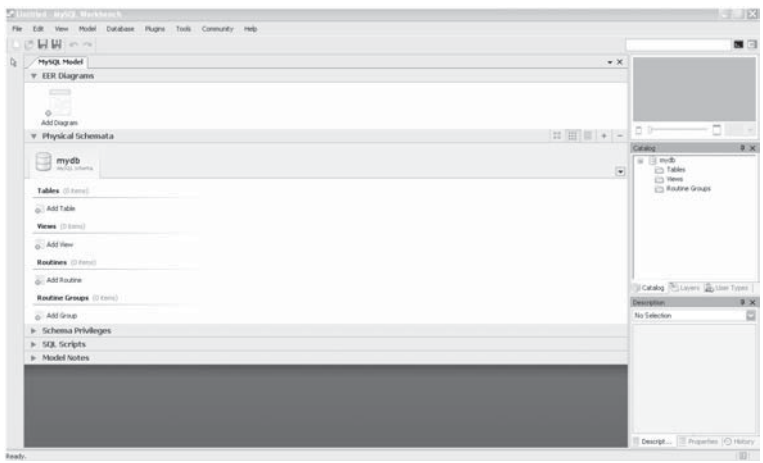


Figura 3-9. MySQL Workbench

MySQL Workbench es una herramienta especialmente orientada al diseño y modelado de bases de datos MySQL. La ventana principal se encuentra dividida en diversas áreas relativas al actual modelo: **diagramas Entidad-Relación extendidos (EER), esquemas físicos, privilegios del esquema, scripts SQL y notas**. Para cada esquema físico de la base de datos podemos ver las **tablas, vistas, rutinas y grupos de rutinas** disponibles.

No sólo podemos visualizarlas, sino que es posible crear nuevas, eliminarlas… Si desplegamos el área privilegios del esquema de la base de datos podemos ver los usuarios y roles definidos para el mismo (véase la figura 3-10).

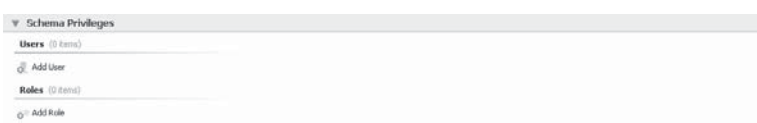


Figura 3-10. Privilegios de una base de datos

Es especialmente útil en el **diseño de bases de datos**. Para ello podemos crear los diagramas Entidad-Relación de una manera sencilla, pudiendo especificar plenamente las características de tablas, campos, claves, relaciones entre tablas, etc. Es simple añadir

y editar posteriormente tablas y relaciones haciendo uso de los botones de la barra de herramientas y los menús correspondientes (véase la figura 3-11).

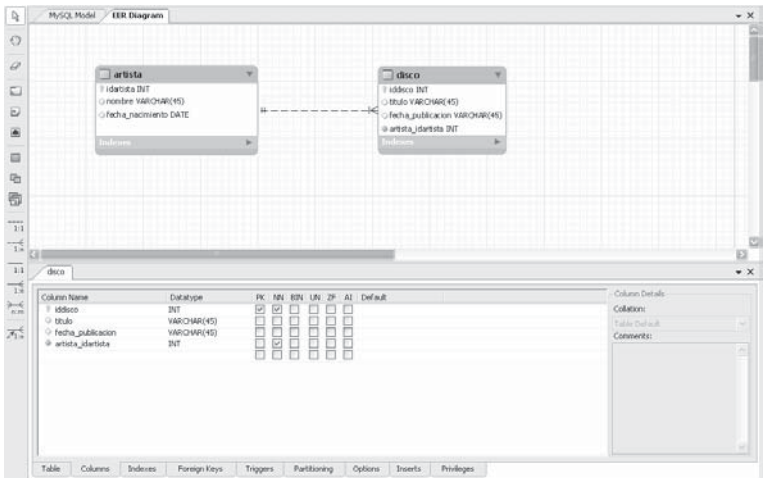


Figura 3-11. Editando el modelo E-R

Para cada tabla es posible especificar, por ejemplo, el motor de la base de datos, el lenguaje, los índices, campos, claves foráneas, incluso triggers o privilegios. Además de poder personalizar al completo tanto la notación en los diagramas EER (menú *Model*) como la propia interfaz gráfica de la aplicación (menú *View*) podemos realizar dos operaciones con gran importancia:

- En el **menú Database** podemos administrar las conexiones de la herramienta y los modelos que hayamos desarrollado con bases de datos MySQL.

  Desde aquí podemos realizar **ingeniería inversa** (obtener esquema a partir de una base de datos), **ingeniería directa** (aplicar el esquema modelado en una base de datos) o incluso **sincronizar el modelo con una determinada base de datos**. Esto es muy importante ya que eligiendo una determinada conexión con una base de datos, podemos seleccionar qué objetos de nuestro modelo copiar en la base de datos y viceversa.

- En el **menú File**, además de las opciones habituales, disponemos de opciones muy interesantes para la **exportación de nuestro modelo en forma de script SQL**, bien de creación, bien de modificación o de sincronización. Estos scripts pueden ser ejecutados en la consola MySQL, lo que nos permite por ejemplo crear de un solo paso toda nuestra base de datos modelada con MySQL Workbench.

  En la creación de estos scripts es posible configurar un sinfín de opciones para determinar las sentencias SQL que queremos incluir: sentencias de creación, eliminación, insert into, claves foráneas… De la misma forma podemos seleccionar si incluir tablas o no, vistas, rutinas, usuarios… En la figura 3-12 se muestra el resultado al generar un script.

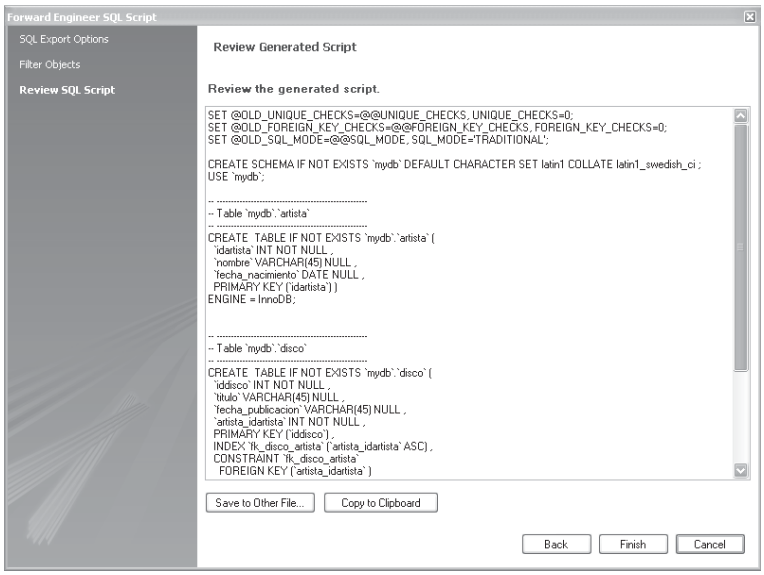


Figura 3-12. Script generado

3.4.4.3 MYSQL FRONT

MySQL Front es una herramienta que permite la gestión y administración de bases de datos MySQL. A diferencia de MySQL Workbench, permite explorar con gran eficiencia y facilidad el esquema y contenido de una base de datos MySQL. Además de visualizar este tipo de información, es posible **diseñar consultas sobre la base de datos, ejecutar scripts SQL, explorar objetos y datos, operar con tablas** (campos, claves), **rutinas, eventos, restricciones, triggers, etc**.

3.4.4.3.1 Instalación en sistemas operativos Windows

A continuación mostramos el proceso de instalación de MySQL Front 5.0 en sistemas operativos Windows:

- Descargamos de la web oficial (http://www.mysqlfront.de/wp/) el archivo de instalación e iniciamos el proceso de instalación.
- Tras aceptar la página de bienvenida al asistente de instalación nos solicita dónde queremos instalar la aplicación y la carpeta del menú de inicio donde queremos realizar la instalación.
- Finalmente, seleccionamos los iconos que queremos crear para el acceso a la aplicación y su integración con el explorador de Windows, para la asociación con la extensión *.sql* y el protocolo *mysql://* (véase la figura 3-13).

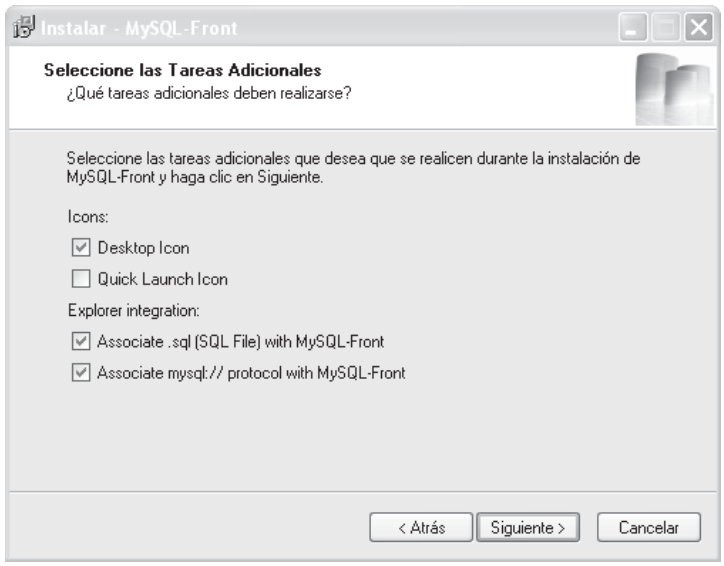


Figura 3-13. Instalación de MySQL - Front

- Finalmente, el sistema nos muestra un resumen de la configuración y pulsamos *Instalar*.

3.4.4.3.2 Entorno gráfico

Una vez que lanzamos MySQL Front, debemos configurar una primera conexión con una base de datos para poder trabajar. Introducimos equipo, puerto, nombre de usuario y contraseña (véase la figura 3-14).

Una vez que hemos configurado el acceso a una base de datos y logramos abrir su esquema podemos proceder a realizar cualquier tipo de tarea administrativa sobre ella. A parte de las ya habituales en este tipo de aplicaciones y que han sido comentadas al principio de este apartado, podemos realizar otras también muy interesantes como transferir información entre diferentes bases de datos, importar y exportar la información de/a muy diversos formatos, configurar las sesiones de usuarios de la base de datos, o manipular diversos servidores con bases de datos pudiendo sincronizar tablas y esquemas entre ellos.

Figura 3-14. Añadir sesión

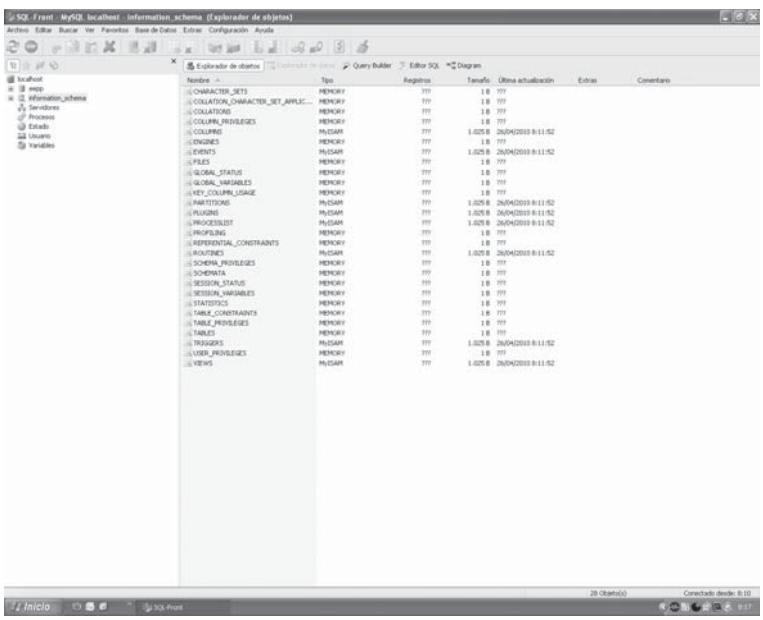


Figura 3-15. MySQL Front

3.5 PHP

PHP (*PHP Hypertext Pre-processor*) es un lenguaje de programación interpretado por el servidor de páginas web de forma que éstas pueden ser generadas de forma dinámica. PHP no sólo es utilizado para este propósito, sino que además puede ser usado

desde una interfaz de línea de comandos o para la creación de aplicaciones con interfaces gráficas.

En la dirección web oficial del proyecto (http://php.net/) puede encontrar una amplia documentación sobre el lenguaje: manuales, sintaxis utilizada, interfaz para la programación de las aplicaciones, etc.

Para instalar PHP automáticamente ejecutamos:

```
# yum install php
```

Para comprobar que PHP ha sido instalado con éxito podemos crear un fichero *php* y ubicarlo en el directorio raíz del servidor web. Por ejemplo para mostrar toda la información útil disponible y detalles sobre la instalación actual de PHP, editamos el fichero */var/www/html/info.php.*

```
# nano /var/www/html/info.php
```

El contenido del fichero incluye una sentencia para ejecutar la función *phpinfo()* que nos permite obtener la información sobre el módulo PHP.

```
<?php
   phpinfo();
?>
```

Así, al ejecutar el fichero en una petición HTTP el servidor lanza la sentencia y muestra el contenido pedido, de forma dinámica. Antes de probar a ejecutar este fichero reiniciamos el servidor Apache:

```
# service httpd restart
```

Ahora sí, abrimos el navegador web y escribimos en la barra de direcciones http://localhost/info.php.

Como podemos ver en la figura 3-16, PHP5 se encuentra correctamente instalado. Si observamos con detenimiento la información mostrada podemos ver por ejemplo que trabaja a través de Apache 2.0 (*línea Server API*), los módulos actualmente habilitados para ser ejecutados con PHP. Por ejemplo, MySQL no aparece listado, por lo tanto en este momento aún no tenemos soporte para MySQL en PHP.

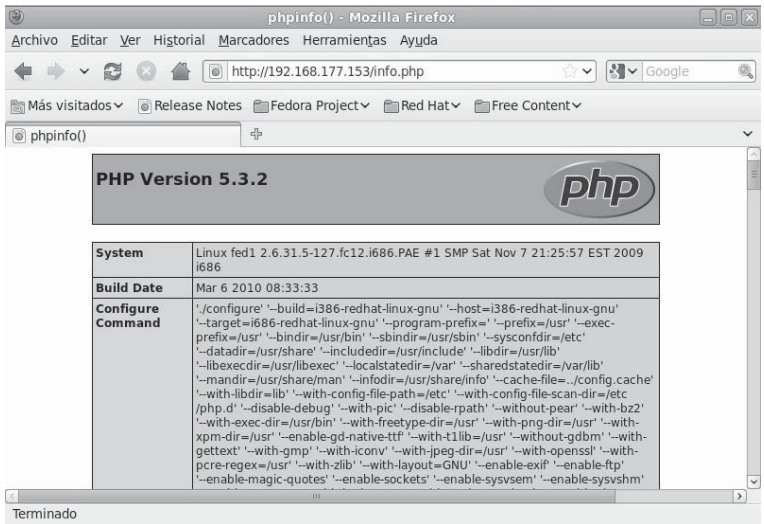


Figura 3-16. PHP – Ejecución de phpinfo()

3.5.1 MySQL y PHP

A continuación vamos a instalar el módulo MySQL para PHP y así aprender cómo es posible cargar diferentes módulos para PHP en función de los requisitos que nos presenten las aplicaciones de nuestro servidor LAMP.

Para dar soporte para MySQL a PHP es necesario instalar el paquete *php-mysql.* Además, aprovechamos la ocasión para instalar otros módulos PHP5 cuyo uso es bastante habitual en aplicaciones web. Para realizar una búsqueda de los módulos disponibles podemos ejecutar:

```
# yum search php
```

Elegimos los que queremos incluir de la lista de salida del comando anterior y ejecutamos *yum install* para instalarlos:

```
# yum install php-mysql php-gd php-imap php-ldap php-odbc php-pear
php-xml php-xmlrpc php-eaccelerator php-magickwand php-magpierss
php-mapserver php-mbstring php-mcrypt php-mssql php-shout php-snmp
php-soap php-tidy
```

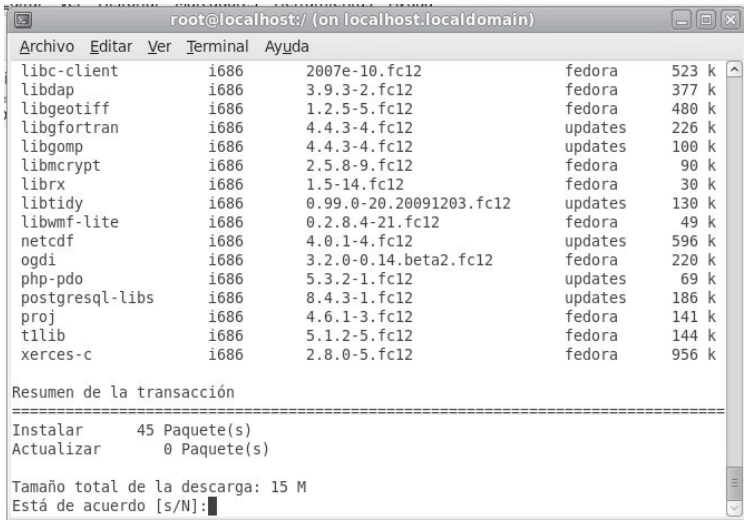


Figura 3-17. PHP – Instalación de módulos

Para que la instalación de los módulos realizada surta efecto debemos reiniciar el servidor web ejecutando:

```
# service httpd restart
```

Si actualizamos la página web *info.php* en el navegador y volvemos a la sección de módulos veremos cómo ahora los módulos instalados aparecen como disponibles para ser utilizados, incluyendo el módulo *mysql* (véase la figura 3-18).

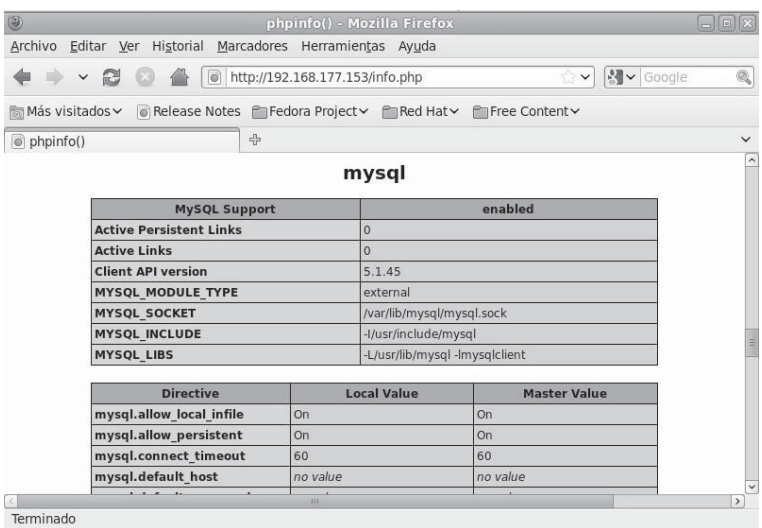


Figura 3-18. PHP – Comprobando el módulo mysql

3.5.2 Modificar el tamaño de subida de ficheros

Para modificar el tamaño de los ficheros que se pueden subir a través del servidor web necesitamos editar el fichero */etc/php.ini* e indicar el tamaño en la variable *post_max_size.*

```
; Maximum size of POST data that PHP will accept.
; http://www.php.net/manual/en/ini.core.php#ini.post-max-size
post_max_size = 500M
```

Para que se apliquen los cambios deberemos recargar la configuración del servidor de páginas web ejecutando:

```
service httpd reload
```

Capítulo 4

PHP

4.1 INTRODUCCIÓN

PHP *(Hypertext Pre-procesador*) es un lenguaje de programación interpretado en el lado del servidor, originalmente diseñado para la creación de páginas web dinámicas, aunque actualmente puede ser usado para crear otro tipo de aplicaciones, incluso con interfaz gráfica. Fue creado por Rasmus Lerdorff, y apareció por primera vez en el año 1995, y desde entonces se ha convertido en uno de los lenguajes más utilizados para el desarrollo de aplicaciones web del lado del servidor.

Normalmente el código PHP se ejecuta en un servidor web, tomando como entrada el propio código PHP y generando como salida la página Web.

Entre sus ventajas podemos destacar:

- Es un lenguaje multiplataforma.
- Es libre.
- Tiene una amplia biblioteca de funciones.
- Posee una amplia documentación.
- Aunque no es un lenguaje orientado a objetos, permite la programación orientada a objetos.

Más información

Sitio oficial de PHP
http://www.php.net/

Más información

En el capítulo 3 dispones de información de cómo instalar el servidor web con el módulo de PHP.

4.2 ELEMENTOS BÁSICOS

Los elementos básicos de PHP tienen gran similitud con los de los lenguajes C y Perl. A continuación se enumeran los principales:

- **Identificadores**. Se utilizarán para nombrar las entidades del lenguaje: variables, constantes, objetos, funciones, etc.
- **Variables**. Es un elemento que puede adquirir o ser sustituido por un valor de cualquier tipo de datos que soporte el lenguaje. Tiene asociada una posición de memoria. Comenzarán siempre por el símbolo $.
- **Tipos de datos**. Son los diferentes tipos de representación de datos que soporta el lenguaje: números enteros, flotantes, cadenas de texto, etc.
- **Palabras reservadas**. Son las palabras utilizadas por el propio lenguaje. Por ejemplo, son utilizadas para definir: los tipos de datos, operadores, estructuras de control, etc.
- **Operadores**. Son los que nos permiten realizar cálculos u operaciones con las variables.
- **Expresiones**. Es cualquier combinación de variables, operadores, o cualquier otro elemento que se puede poner a la derecha del operador de asignación.
- **Estructuras de control**. Nos permiten decidir el flujo de ejecución del programa.
- **Funciones y/o métodos**. Conjunto de expresiones agrupadas que realizan una determinada operación.
- **Comentarios**. Son utilizados para establecer anotaciones en el código fuente, para que sea fácilmente comprendido por terceros.

A continuación vamos a ver con más detalle cada uno de los elementos que forman el lenguaje.

4.2.1 Identificadores

Los identificadores permiten asignar nombres a las variables, objetos, constantes, funciones…, permitiéndonos posteriormente realizar llamadas a los mismos.

En primer lugar, debemos tener presente que PHP es case sensitive (sensible a mayúsculas y minúsculas). Por tanto, no es lo mismo ‘identificador’ que ‘Identificador’.

Para formar un identificador se siguen las siguientes reglas:

- Un identificador debe comenzar por una letra o por el caracter '_'.
- El siguiente caracter, y sucesivos, puede estar formado por cualquier letra, número, o por el caracter '_'.

Para nombrar las variables cambia las reglas de nombrado, ya que se le antepone al identificador el símbolo dólar ($).

Por ejemplo:

```
<?php
$var1         //Identificador válido
$_Var1        //Identificador válido
$Var2         //Identificador válido
%4Var         //Identificador no válido
?>
```

4.2.2 Palabras reservadas

Las palabras reservadas son las palabras utilizadas por el lenguaje, y por tanto no pueden ser utilizadas como identificadores.

En la tabla 4-1 se muestran las palabras reservadas del lenguaje PHP.

Tabla 4-1. Palabras reservadas

Palabras reservadas				
abstract	continue	endforeach	global	switch
array	continue	endif	goto	throw
as	declare	endswitch	if	try
break	default	endwhile	implements	use
break	do	extends	namespace	var
case	else	final	new	while
catch	elseif	for	old_function	xor
clone	enddeclare	foreach	private	
const	endfor	function	protected	

4.2.3 Comentarios

Los comentarios permiten establecer anotaciones descriptivas en el código fuente, para que sea más fácil comprender las distintas partes del programa.

PHP permite los mismos tipos de comentarios que C, C++ y Perl. Los comentarios se pueden escribir de dos formas:

1. **Comentarios de una sola línea.** El comentario debe ir precedido de la doble barra (//) o del caracter #.
2. **Comentarios en varias líneas.** El comentario se encuentra delimitado por los elementos (/* …. */).

A continuación podemos ver un ejemplo de cada tipo de comentario.

```
//Esto es un comentario
#Esto es un momentario
/* Esto
también es un comentario*/
```

4.2.4 Constantes

Una constante es un tipo de dato especial que no cambia de valor durante la ejecución del programa. La definición de una constante se realiza de la siguiente forma:

```
define('NombreConstante', Valor);
```

Como norma general se suelen declarar las constantes en mayúsculas para que sean fácilmente diferenciables de las variables.

A continuación podemos ver un ejemplo de definición de constantes.

```
<?php
      define('PI',3.1416);
      $diametro=2.5;
      $longitud=PI*$diametro;
?>
```

4.3 TIPOS DE DATOS

Los tipos de datos en PHP se pueden dividir en 8 tipos de datos primitivos que se agrupan en 3 categorías: tipos escalares, tipos compuestos y tipos especiales.

A continuación se van a ver las características y el modo de utilización de cada tipo de datos:

4.3.1 Tipos de datos escalares

Los tipos de datos escalares se clasifican en:

- **Integer.** Los números enteros se pueden representar utilizando las notaciones: decimal, hexadecimal y octoal.

A continuación puede ver un ejemplo donde se aprecia la sintaxis de cada notación:

```
<?php
$decimal = 50;         # Decimal
$hexadecimal = 0x32;  # Hexadecimal – Va siempre precedido de
0x
$octal = 062;          # Octal – Va precedido siempre de un 0
?>
```

- **Float.** Se utiliza para representar los números en coma flotante, y se pueden representar utilizando las siguientes notaciones:

```
<?php
$a = 21.144;
$b = 125.252e3;
$c = 1E-10;
?>
```

- **String.** Se utiliza para almacenar cadenas de caracteres y permite diferentes formas de definición de datos:

```
<?php
$cadena = "Hola ";
$cadena = 'hola';
$cadena = <<<'EOD';
?>
```

Se recomienda utilizar la doble comilla ("") para utilizar cadenas de caracteres porque permite definir más caracteres de escape.

- **Booleano.** Es un tipo de datos simple que sólo tiene dos posibles valores o estados: *True* o *False*.

```
<?php
  $valor = True;
?>
```

4.3.2 Tipos de datos compuestos

PHP permite utilizar dos tipos de datos compuestos:

- **Array.** Permite utilizar un conjunto de valores en una sola variable y acceder a ellos a través de un índice. PHP permite que el índice pueda ser un número entero o una cadena de caracteres.

En el siguiente ejemplo podemos ver dos ejemplos de definición y utilización de arrays.

```
<?php
# Definicion de un array
$ar1 = array(20,32,45,3,90);
```

```
print ar1[0;         // Muestra 20
print ar1[3];        // Muestra 45


# Definicion de un array
$ar2 = array('color' => 'rojo', 'marca' => 'bmw',
'cilindrada' => 2.5 );
print ar2[color];    // Muestra rojo
print ar2[marca];    // Muestra bmw
?>
```

- **Objeto.** Un objeto está compuesto por un conjunto de variables y funciones. A estas variables y funciones accederemos a través del operador (->). Veamos un ejemplo:

```
<?php
 class cliente{      //Definición del objeto
public $nombre;
public $apellido;
public $DNI ;
 }

$cli = new cliente;         // Instancia del objeto
$cli->nombre = "Jose";      // Establece el valor de una
variable del objeto
?>
```

4.3.3 Tipos de datos especiales

Por último, PHP define dos tipos de datos especiales:

- **Null.** Utilizado para indicar el vacío, es decir, para especificar que a una variable no se le ha asignado ningún valor.

- **Resource.** Utilizado para contener referencias a recursos externos, como por ejemplo funciones especiales.

4.4 OPERADORES

A continuación, veremos los distintos operadores agrupados por su funcionalidad.

4.4.1 Operadores de asignación

Los primeros operadores que vamos a presentar, y los más importantes, son los operadores de asignación, ya que permiten asignar valores a las variables.

Tabla 4-2. Operadores de asignación

Símbolo	Descripción
=	Asigna un valor a una variable
+=	Operador de suma asignación compacta
-=	Operador de resta asignación compacta
*=	Operador de multiplicación asignación compacta
/=	Operador de división asignación compacta
%=	Operador de módulo asignación compacta
.=	Operador de concatenación asignación de cadenas

En el siguiente ejemplo, podemos ver un ejemplo de uso, alternativas y equivalencias de los operadores de asignación.

```
<?php
$a = 10;
$b = 5;
$a+=$b;       // $a= $a + $b;     Resultado  a = 15
$a-=$b;       // $a= $a - $b;     Resultado  a = 5
$a*=$b;       // $a= $a * $b;     Resultado  a = 50
$a/=$b;       // $a= $a / $b;     Resultado  a = 2

$c=”hola”;
$d=”mundo”;
$c.=$d;       // $c= $c . $d;     Resultado  c = “holamundo”
?>
```

4.4.2 Operadores aritméticos

Los operadores aritméticos permiten realizar operaciones matemáticas simples (p.e. suma, resta, multiplicación). En la tabla 4-3 se muestran los diferentes tipos de operadores aritméticos.

Tabla 4-3. Operadores aritméticos

Símbolo	Descripción
+	Suma
-	Resta
*	Multiplicación
/	Dividir
%	Resto de la división
++	Incremento
--	Decremento

A continuación podemos ver un ejemplo de uso de los operadores aritméticos:

```
<?php
$b=50;
$a=10;
```

```
$a+$b;        // Resultado 60
$a++;         // Resultado 11
?>
```

4.4.3 Operadores de comparación

Los operadores de comparación permiten comparar dos o más elementos devolviendo *True* si el resultado es verdadero y *False* en caso contrario. En la tabla 4-4 se pueden ver los diferentes operadores de comparación.

Tabla 4-4. Operadores de comparación

Símbolo	Descripción
==	Igual
!=	Distinto o diferente
<>	Distinto o diferente
>=	Mayor o igual
<=	Menor o igual
>	Mayor que
<	Menor que
===	Idéntico
!==	No idéntico

Podemos entender mejor su funcionamiento viendo el siguiente ejemplo.

```
<?php
$hoy = 4;
$ayer = '4';
$comp = $hoy == $ayer;    // comp adquiere el valor true
$comp = $hoy === $ayer    // comp adquiere el valor false
?>
```

4.4.4 Operadores lógicos

Estos operadores permiten realizar operaciones lógicas entre dos o más valores y proporcionan un valor de salida binario (*True* o *False*).

Tabla 4-5. Operadores lógicos

Símbolo	Descripción
&& **AND**	AND (Y lógico). Si la primera expresión no es cierta no evalúa la segunda
|| **OR**	OR (O lógico). Si la primera expresión es cierta no evalúa la segunda
XOR	XOR (O exclusiva). Devuelve verdadero si y solo si uno de los operandos es verdadero, en caso contrario falso
!	NOT (No lógico)

A continuación podemos ver un ejemplo:

```
<?php
$a = True;
$b = False;
$a && $b;           //Devuelve el valor falso
$a XOR %b;          //Devuelve el valor verdadero
?>
```

4.4.5 Operadores binarios

Los operadores binarios permiten realizar las operaciones a nivel de bit.

Tabla 4-6. Operadores binarios

Símbolo	Descripción
~	Complementación
<<	Desplazamiento a la izquierda
>>	Desplazamiento a la derecha
&	AND (Y Binario). Bit a bit
|	OR (O Binario). Bit a bit
^	XOR. Bit a bit

A continuación podemos ver un ejemplo:

```
<?php
$a = 12;
$b = ~$a;     //El resultado de la complementación es 3
              //12          1100  ~  0011              3
?>
```

4.4.6 Operadores de cadena

Este operador (.) permite concatenar dos cadenas de caracteres. A continuación se muestra un ejemplo de uso:

```
<?php
$a = "Hola ";
$b = "Mundo";
print $a.$b;        // Muestra por pantalla Hola Mundo
?>
```

4.5 ESTRUCTURAS DE CONTROL

Las estructuras de control permiten cambiar el flujo del programa, en función del estado de las variables.

4.5.1 Condicionales simples (if)

Las condiciones simples permiten ejecutar una parte del código cuando se cumple una determinada condición.

La instrucción *if* es la estructura básica de las estructuras de control. Se encarga de evaluar la expresión booleana, y en el caso de que sea verdadera, ejecutar la instrucción o grupos de instrucciones que la precedan, opcionalmente puede ir seguida de la instrucción *else*, que proporciona una instrucción o grupo de instrucciones a ejecutar, en el caso de que no sea verdadera la expresión boolena.

```
<?php
	$a = 5;
	$b = 6;
	if($a < $b)
		echo "A es mayor que B";
	else
		echo "A es menor o igual que B";
?>
```

El operador ternario es una instrucción muy parecida a la instrucción if else, aunque cambia la sintaxis.

```
expresión ? instrucción1: instrucción2;
```

A continuación se muestra el código alternativo al ejemplo anterior, pero haciendo uso del operador ternario en lugar de la instrucción *if.*

```
<?php
	($a < $b)? echo "A es mayor que B" : echo printf "A es menor
o
	igual que B";
?>
```

4.5.2 Selección múltiple

En determinadas situaciones nos encontramos con que tenemos múltiples posibilidades de selección como resultado de evaluar una expresión. Existen dos formas de permitir la selección múltiple.

4.5.2.1 IF ANIDADO

Una forma de realizar condiciones de selección múltiple es utilizar la sentencia *if* de forma anidada. A continuación podemos ver un ejemplo en que se toma una decisión dependiendo del valor de la variable *color*.

```
<?php
	color="azul";

	if ($color == "rojo" )
	{
```

```
            echo "El color es rojo";
      }
      else if ($color == "verde")
      {
            echo "El color es verde";
      }
      else if($color == "azul" )
      {
            echo "El color es azul";
      }
?>
```

Tal y como podemos observar en el ejemplo, conforme vamos aumentando el número de posibilidades de selección, el código se va haciendo cada vez más complejo y difícil de seguir. Ante estas situaciones se recomienda el uso de la instrucción *switch*.

4.5.2.2 SWITCH

Switch permite evaluar una expresión y realizar diferentes operaciones dependiendo de su valor. Su sintaxis es:

```
switch ( Expresión )
{
      case Valor1:
            Instrucción o bloque de instrucciones;
            [break;]
      case Valor2:
            Instrucción o bloque de instrucciones;
            [break;]
      ...
      case Valor N:
            Instrucción o bloque de instrucciones;
            [break;]
      [default:]
            Instrucción o bloque de instrucciones;
}
```

A continuación se muestra el código alternativo con la instrucción *switch*, al *if* anidado, para poder observar las diferencias.

```
<?php
$color= rojo;
switch ($color)
 {
      case "rojo":
            echo "El color es rojo";
            break ;
      case "verde":
            echo "El color es verde";
            break ;
      case "azul":
            echo "El color es azul";
            break;
}
?>
```

4.5.3 Bucles

Los bucles permiten repetir una o varias instrucciones un número determinado de veces. Existen dos tipos de bucles: los que se repiten un número determinado de veces (*for*) o los que se repiten hasta que se cumpla una determinada condición de salida (*while* y *do while*).

4.5.3.1 EL BUCLE FOR

El bucle for se utiliza para ejecutar una instrucción o grupo de instrucciones un número determinado de veces. Su sintaxis es muy parecida a la sintaxis del bucle en C o Java, la podemos descomponer en tres partes:

- **Inicialización**. Declara el valor inicial de la variable contador.
- **Condición**. Es donde se comprueba que se cumple la condición de ejecución.
- **Actualización**. Indica cómo actualizar la variable contador.

```
   Sintaxis:
for ([inicialización]; [condición]; [actualización] )
{
       Instrucción o grupo de instrucciones;
}
```

A continuación, se muestra un ejemplo de un bucle for, que visualiza por pantalla los números del 0 al 100.

```
<?php
for($i=0;$i<=100;$i++)
{
       Echo $i."<BR>";
   }
?>
```

PHP posee una variante bastante útil del bucle for, el bucle *foreach*. Permite iterar por los distintos elementos de un array. Por lo tanto, el bucle se ejecutará tantas iteraciones como elementos tenga el array.

```
foreach (variable as valor)
{
       Instrucción o grupo de instrucciones;
}
```

Con el siguiente ejemplo podemos entender su funcionamiento. Por ejemplo, podemos observar que sin colocar ningún índice, se recorre todo el array.

```
<?php
$cestaCompra= array("pan","tomates","jamón");

echo "El contenido de la cesta de la compra es:<BR>";
foreach ($cestaCompra as $i)
```

```
{
            echo $i."<BR>";
}
?>
```

4.5.3.2 EL BUCLE WHILE

El bucle *while* se utiliza cuando se desea que una instrucción o grupo de instrucciones se ejecute un número indefinido de veces, mientras se cumpla una determinada condición. Su sintaxis es mucho más sencilla que la del bucle *for*, ya que se compone de la palabra reservada *while* y la condición de ejecución entre paréntesis.

```
while (condición)
{
   Instrucción o grupo de instrucciones;
}
```

4.5.3.3 EL BUCLE DO WHILE

El bucle *do while* es muy parecido al bucle *while*, con la diferencia de que *do while* al menos se ejecuta siempre una vez, y no sabemos cuántas veces se va a ejecutar.

Con el bucle *do while* primero se ejecuta la instrucción o grupo de instrucciones y después se comprueba la condición de ejecución.

```
do
{
       Instrucción o grupo de instrucciones;
}while (condición);
```

4.5.4 Instrucciones break y continue

La sentencia *break* se utiliza para romper la ejecución del bucle donde se encuentra y saltar a la siguiente instrucción.

Esta sentencia no es muy habitual encontrarla en programación, a excepción de cuando se usa junto con la instrucción *switch*, ya que rompe la secuencia de programa y hace difícil y costoso la depuración de los programas.

La instrucción *continue* al igual que la anterior *break*, son muy parecidas y poco recomendables de utilizar. *Continue* se utiliza para romper la ejecución de una iteración de un bucle y seguir con la ejecución de la siguiente iteración, impidiendo que se ejecuten las instrucciones que le sucedan dentro del bucle.

4.6 FUNCIONES

Una función es un bloque de código que permite su reutilización de una forma fácil y sencilla. Se recomienda que el nombre de la función sea lo más descriptivo posible y que describa lo más fielmente posible el funcionamiento de la función.

4.6.1 Definición de funciones

Para definir una función se hace a través de la palabra reservada *function*, seguida del nombre utilizado como identificador de la función. A continuación, se declara la instrucción o conjunto de instrucciones que se ejecutarán cada vez que se realiza la llamada a la función, como siempre entre llaves.

Opcionalmente las funciones pueden aceptar uno o más valores de entrada, conocidos como parámetros de la función. Éstos son introducciones separadas por comas y entre paréntesis tras el nombre de la función. También y opcionalmente la función puede devolver un valor de retorno, para ello, se hará uso de la palabra clave *return*.

```
function nombre ([arg1,arg2,…,argN])
{
       Instrucción o grupo de instrucciones;
       [return var;]
}
```

Como ejemplo, vamos a crear una función que acepte dos valores de entrada y nos devuelva la suma de los mismos. En el ejemplo podemos ver la definición y uso de la función.

```
<?php
function sumar($a, $b)
{
   return $a + $b;
}
?>
<HTML>
       <HEAD>
              <TITLE>
                     Funciones en php
              </TITLE>
       </HEAD>
       <BODY>
       <?php echo sumar(3,2);?>
       </BODY>
</HTML>
```

4.6.2 Funciones predefinidas

PHP, a parte de permitir crear nuestras propias funciones, nos proporciona una librería de funciones bastante extensa, que pueden ser de gran utilidad. Ya que, como se comentó en la introducción es una de sus ventajas competitivas.

A continuación se va a mostrar un resumen de la librería de funciones de PHP oficial, ordenadas por funcionalidad.

- Manipulación de formatos de audio.
- Servicios de autenticación.
- Extensiones relacionadas con fecha y hora.
- Compresión y extensiones de archivos.
- Proceso de tarjetas de crédito.

- Extensiones criptográficas.
- Extensiones de base de datos.
- Extensiones relacionadas con el sistema de ficheros.
- Procesado y generación de imágenes.
- Extensiones relacionadas con email.
- Extensiones matemáticas.
- Control de procesos.
- Extensiones de sesión.
- Procesado de textos.
- Manipulación XML.

A modo de ejemplo, en la tabla 4-7 se muestran las funciones para el manejo de cadenas de caracteres.

Tabla 4-7. Funciones para el manejo de cadenas

Función	Descripción
count_chars(cadena,modo)	Número de repeticiones de un caracter en una cadena.
chop(cadena)	Obtiene la cadena de caracteres de entrada sin espacios ni saltos de línea.
echo	Muestra el contenido de la cadena por pantalla.
ltrim(cadena)	Elimina los espacios que aparecen a la derecha de una cadena de caracteres.
ord(cadena)	Devuelve el valor ASCII de un caracter.
printf ()	Muestra una cadena de caracteres.
rtrim(cadena)	Elimina los espacios que aparecen por la derecha en una cadena de caracteres.
sprintff ()	Es igual que *printf* pero guarda la salida en una variable.
str_replace (subcadena1,subcadena2,cadena)	Sustituye una palabra por otra dentro de una cadena.
strcasecmp(cadena1,cadena2)	Compara dos cadenas sin ser sensible a mayúsculas y minúsculas.
strcmp(cadena1,cadena2)	Compara dos cadenas siendo sensible a mayúsculas y minúsculas.
strlen(cadena)	Indica el número de caracteres de una cadena.
strnatcmp(cadena1,cadena2)	Compara dos cadenas de caracteres y es sensible a mayúsculas y minúsculas.
strncmp(cadena1,cadena2,tamaño)	Compara los N primeros caracteres de una cadena.
strpos(cadena,subcadena)	Obtiene la primera ocurrencia de una cadena en otra.
strrpos(cadena,subcadena)	Obtiene la última ocurrencia de una cadena en otra.
strstr(cadena,subcadena)	Localiza una subcadena dentro de la cadena original.
strtolower(cadena)	Convierte la cadena en minúsculas.
strtoupper(cadena)	Convierte la cadena en mayúsculas.
trim(cadena)	Elimina los espacios que aparecen a izquierda y derecha de la cadena.
ucfirst(cadena)	Convierte en mayúscula el primer caracter de la cadena.
ucwords(cadena)	Convierte en mayúsculas el primer caracter de cada palabra de la cadena.

Nota

Antes de implementar una función es recomendable que busques si ya está implementada en alguna librería.

Más información

En la web oficial www.php.net puedes encontrar información sobre las diferentes librerías que hay implementadas en PHP.

4.7 VARIABLES PREDEFINIDAS

PHP proporciona una gran cantidad de variables predefinidas a cualquier script que se ejecute. Estas variables son de sólo lectura, por tanto no pueden ser modificadas. Muchas de estas variables dependen del servidor donde se está ejecutando (de la versión, configuración y otros factores), por lo que puede que algunas de las que se describen no estén disponibles. Podemos ejecutar el siguiente código y ver todas las que tenemos disponibles para nuestra instalación.

```
<HTML>
   <?php
      phpinfo();
   ?>
</HTML>
```

En la figura 4-1 se muestra el resultado del código anterior.

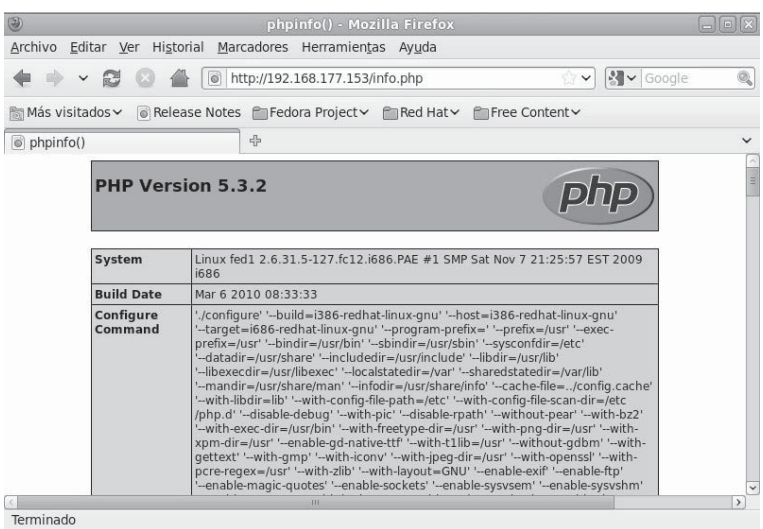


Figura 4-1. PHP – Ejecución de phpinfo()

En la tabla 4-8 podemos ver las variables y su utilidad.

Tabla 4-8. Variables predefinidas

Función	Descripción
$HTTP_ACCEPT_LANGUAGE	Idioma del navegador
$HTTP_REFERER	Dirección de la URL de procedencia del usuario. Con ella podemos ver la última página que visitó.
$HTTP_USER_AGENT	Navegador que está utilizando el usuario
$OS	Versión del sistema operativo donde se ejecuta la página
$REMOTE_ADDR	IP del usuario
$REMOTE_HOST	Nombre del host desde el que se está viendo la página.
$REQUEST_METHOD	Método de petición que fue usado para acceder a la página.
$SERVER_NAME	Nombre del servidor donde se ejecuta el script

En el siguiente ejemplo podemos ver cómo se han utilizado las variables predefinidas para mostrar información al usuario.

```
<HTML>
	<HEAD>
		<TITLE>
			Variables Predefinidas
		</TITLE>
	</HEAD>
	<BODY>
	<?php
		echo "Tu dirección IP es: ";
		echo $_SERVER['REMOTE_ADDR'];
		echo "<BR>";
		echo "Estas usando el navegador: ";
		echo $_SERVER['HTTP_USER_AGENT'];
		echo "<BR>";
	?>
	</BODY>
</HTML>
```

4.8 REUTILIZACIÓN DE CÓDIGO

PHP permite la reutilización de código (scripts), tanto PHP como HTML, incluyéndolo en determinadas partes del programa. Para ello nos suministra las funciones: *require()* y *include()*, que tienen la misma funcionalidad con la única diferencia de que si el archivo llamado no existe, *include()* sólo da una advertencia, y sigue ejecutando el código, mientras que *require()* produce un error e interrumpe la ejecución.

Con estas instrucciones sólo es necesario hacer referencia al fichero, de esta manera, cuando se ejecute la página web, la llamada a la función solicita el fichero y procesa toda la información que contiene.

Uno de los ejemplos de uso más sencillo es el de incluir cabeceras y pies de páginas desde un archivo de plantilla. También es muy utilizado para la utilización de variables globales.

En el siguiente ejemplo podemos ver cómo son usadas estas funciones.

```
<HTML>
       <HEAD>
              <TITLE> Ejemplo de require e include </TITLE>
       </HEAD>
       <BODY>
       <?php
      /*Realizamos la llamada al archivo de variables globales
        en caso de no encontrarlo u otro fallo se suspendería la
        ejecución, evitando que se muestren datos no válidos
       */
      require("variablesGlobales.php");

       /*Sin embargo, en este caso si la cabecera no se carga, no
         provoca un fallo, por tanto no sería crítico para nuestro
         diseño y se permite con la ejecución
       */
       include("cabecera.html");

       ?>
       </BODY>
</HTML>
```

4.9 PHP Y MYSQL

PHP tiene una gran cantidad de librerías que permiten conectar con la mayoría de los gestores de base de datos. A continuación nos vamos a centrar en el gestor de base de datos MySQL por ser el utilizado por la mayoría de los desarrolladores web y ser open source.

Para acceder a una base de datos desde PHP, debemos tener presentes los siguientes pasos:

4.9.1 Instalación de php-mysql

Para poder utilizar las bases de datos MySQL en PHP es necesario instalar el paquete *php-mysql,* por lo que ejecutamos el siguiente comando:

```
# yum install php-mysql
```

Más información

En el capítulo 4 dispones de información de cómo instalar el servidor web con el módulo de PHP y con MySQL.

4.9.2 Conexión con el servidor MySQL

Para conectarse a un gestor de base de datos se utiliza la función *mysql_connect* a la que le debemos indicar la dirección IP o el nombre del servidor de base de datos, el

nombre de usuario y su contraseña. En el siguiente ejemplo, se muestra cómo se realiza la conexión con el servidor, en el caso de que la conexión no se establezca se muestra un mensaje de error.

```
<?php
   $conexion = mysql_connect("localhost","usuario","contrasena")
   or die("Imposible conectar a la base de datos");

   echo "Se ha conectado correctamente";

   mysql close($conexion);
?>
```

4.9.3 Seleccionar una base de datos

Una vez que se ha efectuado la conexión con el servidor, el siguiente paso es seleccionar la base de datos con la que trabajar. Para esta función, PHP proporciona la función *mysql_select_db(),* que devuelve *True* en caso de que la operación se efectúe con éxito, recibiendo por parámetro el nombre de la base de datos con la que se desea conectar.

```
<?php
   mysql select db("miBD", $conexion)
   or die("Imposible consultar la base de datos");

   echo 'Se ha conectado correctamente';
?>
```

4.9.4 Consultas SQL

Tras conectar y seleccionar la base de datos, ya podemos realizar consultas SQL. Para realizar las consultas se utiliza la función *mysql_query (InstrucciónSQL, Enlace)*. Esta función envía la consulta SQL que se encuentra en *InstrucciónSQL* a la base de datos activa asociada al identificador *Enlace.* Si no es especificado un *Enlace* se asume el último enlace abierto. Esta función también nos devolverá *False*, en caso de no tener permiso de acceso a la tabla.

En el siguiente ejemplo podemos ver cómo se usa la función.

```
<?php
    //Preparación de la consulta de selección
    $consulta = "SELECT * FROM CLIENTES;";

    //Envío de la consulta
   $res = mysql query($consulta,$conexion) or die("consulta
incorrecta");
?>
```

4.9.5 Obtener y procesar los resultados

Para obtener y procesar los datos principalmente nos vamos a encontrar con dos funciones: *mysql_num_rows(), mysql_fetch_array().*

- *mysql_num_rows()*. Devuelve el número de filas de una consulta (sólo válido para consultas SELECT), en caso de consultas de inserción, modificación… se debe usar *mysql_affected_rows()*.

```
<?php
  $nfilas = mysql num rows ($res);
?>
```

- *mysql_fetch_array().* Extrae la fila del resultado como una matriz asociativa, una matriz numérica o ambas. En el siguiente ejemplo, se puede ver cómo se extraen las filas de la consulta, y cómo se accede a los datos.

```
<?php
   for ($i=1; $i<$nfilas; $i++)
   {
       $fila = mysql_fetch_array ($res);
       echo "Campo 1: ". $fila["campo1"];
       echo "Campo 2: ". $fila["campo2"];
   }
?>
```

4.9.6 Cerrar la conexión con el servidor

Para finalizar, debemos cerrar la conexión con el servidor para que ésta no permanezca activa un tiempo innecesario. Para ello se utiliza la función *mysql_close(bd)* que cierra la conexión asociada al identificador *bd*.

```
<?php
       mysql_close($conexion);
?>
```

4.10 EJEMPLO

Para ver mejor la integración de PHP y MySQL vamos a ver un ejemplo simple en el que se va a crear el sitio web de un concesionario que tiene una base de datos con los vehículos que vende. Para gestionar dicha base de datos crearemos las operaciones típicas de inserción, modificación y eliminación de vehículos.

En el siguiente ejemplo vamos a suponer que la página web y el servidor MySQL se encuentran en el mismo equipo por lo que las conexiones y permisos se establecerán al equipo *localhost*.

Tal y como podemos ver en la figura 4-2, el sitio web está compuesto por las siguientes páginas:

- **index.php.** Es la página principal donde se muestran todos los vehículos y tenemos los enlaces para modificar, añadir o eliminar un vehículo.
- **anadir.php.** Es el formulario donde podemos introducir los datos del nuevo vehículo. Al pulsar el botón *Enviar* los datos se envían a la página *anadir_guardar.php* que recoge los datos del formulario, los inserta en la base de datos y vuelve a la página principal.
- **modificar.php.** Este formulario es igual que el de la página *anadir.php* pero con la salvedad de que muestra los datos existentes del producto que queremos modificar. Al pulsar el botón *Enviar* los datos se envían a la página *modificar_guardar.php* que recoge los datos del formulario, los inserta en la base de datos y vuelve a la página principal.
- **eliminar.php.** Esta página elimina un determinado vehículo y vuelve a la página principal.

Lógicamente, en las páginas *modificar.php* y *eliminar.php* es necesario conocer el vehículo que se quiere modificar o eliminar respectivamente. Para ello, a través de los enlaces se enviará el campo clave del vehículo, que en nuestro caso es el modelo. De esta forma, la página obtiene el campo clave y puede modificar o eliminar el vehículo correcto.

Para facilitar el acceso a la base de datos y evitar repetir en todas las páginas los comandos para conectarnos y seleccionar la base de datos se crea la página *conexion.php*. Esta página, que será importada por las demás páginas, se conecta y selecciona la base de datos del concesionario.

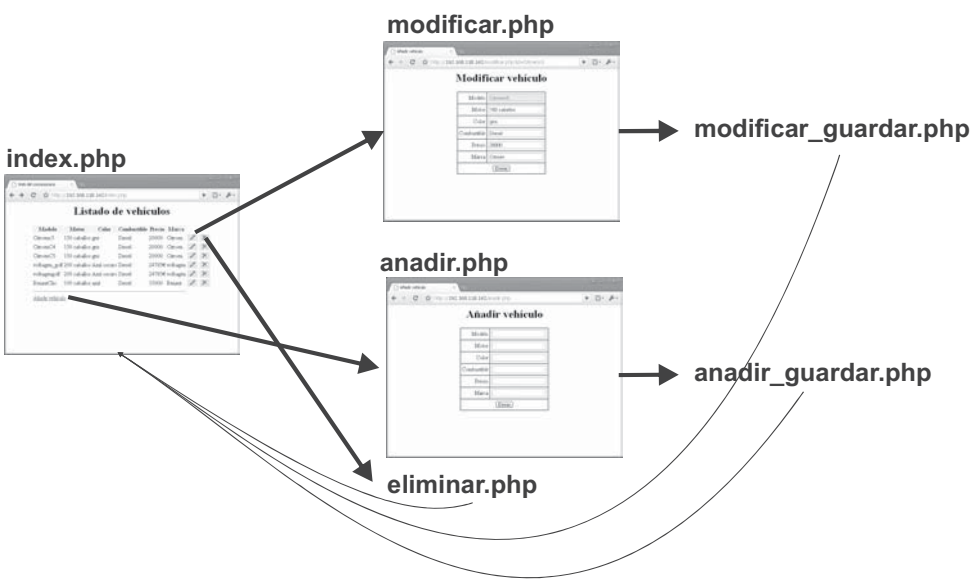


Figura 4-2. Esquema de las páginas web

4.10.1 Creación de la base de datos

Para empezar debemos acceder al terminal de MySQL y crear la base de datos ejecutando:

```
# mysql –u root –p
mysql> create database concesionario;
```

Seleccionamos la base de datos:

```
mysql> use concesionario;
```

Creamos la tabla vehiculos con sus respectivos campos:

```
mysql> CREATE TABLE vehiculos (Modelo VARCHAR(20), Motor
VARCHAR(20), Color VARCHAR(20), Combustible VARCHAR(20), Precio
VARCHAR(20), Marca VARCHAR(20));
   Query OK, 0 rows affected (0.07 sec)
```

Para que existan valores de prueba podemos realizar varias consultas de inserción de la siguiente forma:

```
mysql> INSERT INTO vehiculos (Modelo, Motor, Color, Combustible,
Precio, Marca) VALUES ('Peugeot207', '1299CV', 'Rojo', 'Diesel',
'20000€', 'Peugeot');
   Query OK, 1 row affected (0.04 sec)
```

Y permitimos que un usuario pueda utilizar la base de datos:

```
mysql> GRANT ALL PRIVILEGES ON concesionario.* TO usuario@localhost
IDENTIFIED BY 'contrasena';
```

4.10.2 Conexión a la base de datos

Como vamos a crear un sitio web que tiene varias páginas web que se conectan a la base de datos para insertar, mostrar, modificar o borrar información, vamos a crear una página web que se conecta a la base de datos. De esta forma, en las demás páginas para utilizar la base de datos tan sólo tendremos que incluir la página *conexión.php* y tendremos la conexión en la variable *conexion*.

conexión.php

```
  <?php
    //Conexión con la base de datos
   $conexion = mysql connect("localhost","usuario","contrasena")
   or die("Imposible conectar a la base de datos");

    //Selección de la base de datos
   mysql select db("concesionario", $conexion)
   or die("Imposible consultar la base de datos");
?>
```

4.10.3 Web para mostrar los productos

Una vez que ya está lista la base de datos, en el siguiente código se muestra un ejemplo de utilización mediante una página en PHP:

index.php

```
<HTML>
  <HEAD>
   <TITLE>Web del concesionario</TITLE>
  </HEAD>
  <BODY>
  <?php
    //Nos conectamos a la BD
    include ('conexion.php');

    //Preparación de la consulta de selección
    $consulta = "SELECT * FROM vehiculos;";

    //Envío de la consulta
    $res = mysql query($consulta,$conexion) or die("consulta
incorrecta");

    //Procesamiento de los datos
    $n filas = mysql num rows ($res);

    //Generación de código HTML
    echo "<center><h1> Listado de veh&iacute;culos </h1></center>";
    echo "<table aling=center>\n";
    echo "<tr bgcolor=#ffffaa>
        <th>Modelo</th>
        <th>Motor</th>
        <th>Color</th>
        <th>Combustible</th>
        <th>Precio</th>
        <th>Marca</th>
    </tr>\n";

    for($i=1; $i<=$n filas; $i++)
    {
        $fila = mysql fetch array($res);
        echo "<tr>\n";
        echo "  <td>".$fila['Modelo']."</td>\n";
        echo "  <td>".$fila['Motor']."</td>\n";
        echo "  <td>".$fila['Color']."</td>\n";
        echo "  <td>".$fila['Combustible']."</td>\n";
        echo "  <td>".$fila['Precio']."</td>\n";
        echo "  <td>".$fila['Marca']."</td>\n";
        echo "  <td><a
href=modificar.php?id=".$fila['Modelo']."><img src=ico_modificar.png
border=0></a>";
        echo "  <td><a href=eliminar.php?id=".$fila['Modelo']."><img
src=ico eliminar.png border=0></a>";
        echo "</tr>\n";
    }
    echo "<tr><td colspan=7> <hr>";
    echo "<a href=anadir.php>A&ntilde;adir veh&iacute;culo </a>";
    echo "</td></tr></table>";
    //Se libera la memoria
```

```
    mysql_free_result($res);

    //Se cierra la conexión
    mysql_close($conexion);

       ?>
  </BODY>
</HTML>
```

En la figura 4-3 se muestra el resultado de la ejecución del ejemplo anterior.

Figura 4-3. index.php

Nota

En la página web del libro puedes descargar todo el código del ejemplo anterior.

4.10.4 Formulario para insertar

A parte de mostrar los datos, vamos a permitir al usuario dar de alta vehículos nuevos en la página web. Para realizar la inserción se van a generar dos páginas: *anadir.php,* fichero que contiene el formulario de petición de datos y al pulsar el botón *Enviar* se envían los datos a la página *anadir_guardar.php,* donde se insertan los datos del vehículo y vuelve a la página principal.

Figura 4-4. anadir.php

A continuación podemos ver el código fuente de las páginas *anadir.php* y *anadir_guardar.php.*

anadir.php

```
 <HTML>
  <HEAD>
   <TITLE>Web del concesionario</TITLE>
  </HEAD>
  <BODY>
   <center><h1> A&ntilde;adir veh&iacute;culo </h1></center>
  <form method="get" action="anadir guardar.php">
        <table border="1" align="center" cellpadding="3"
cellspacing="0">
          <tr>
                <td><div align="right">Modelo</div></td>
                <td><input name="Modelo" type="text"></td>
          </tr>
          <tr>
                <td><div align="right">Motor</div></td>
                <td><input name="Motor" type="text"></td>
          </tr>
          <tr>
                <td><div align="right">Color</div></td>
                <td><input name="Color" type="text" ></td>
          </tr>
          <tr>
                <td><div align="right">Combustible</div></td>
                <td><input name="Combustible" type="text">
                </td>
          </tr>
          <tr>
                <td><div align="right">Precio</div></td>
                <td><input name="Precio" type="text" ></td>
```

```
          </tr>
          <tr>
                <td><div align="right">Marca</div></td>
                <td><input name="Marca" type="text" ></td>
          </tr>
          <tr>
                <td colspan=2>
               <div align="center">
                    <input type="submit" name="Submit"
value="Enviar" />
                  </div>
             </td>
          </tr>
        </table>
  </form>
</BODY>
```

anadir_guardar.php

```
 <HTML>
  <HEAD>
   <TITLE>Web del concesionario</TITLE>
  </HEAD>
  <BODY>

<?php
include ('conexion.php');

   // Obtengo las variables
   $Modelo =$_REQUEST["Modelo"];
   $Motor = $_REQUEST["Motor"];
   $Color = $_REQUEST["Color"];
   $Combustible = $_REQUEST["Combustible"];
   $Precio = $_REQUEST["Precio"];
   $Marca = $_REQUEST["Marca"];

    $consulta = "INSERT INTO vehiculos (Modelo, Motor, Color,
Combustible, Precio, Marca) VALUES ('$Modelo', '$Motor', '$Col$
    $res=mysql_query($consulta,$conexion) or die("consulta
incorrecta");

    header("Location:index.php");

?>
```

4.10.5 Formulario para modificar

El funcionamiento para modificar es muy parecido a la inserción de datos salvo que debemos conocer el registro que vamos a modificar para mostrar sus valores actuales en el formulario. Y una vez modificados los datos, se envían a la página *modificar_guardar.php* donde se actualizan los datos en la base de datos y vuelve a la página principal.

Para conocer el registro que debemos modificar se pasa a través del enlace de modificar la variable *modelo*, que en nuestro caso es el campo clave.

Figura 4-5. anadir.php

A continuación podemos ver el código fuente de las páginas *modificar.php* y *modificar_guardar.php*.

modificar.php

```
<HTML>
  <HEAD>
   <TITLE>Web del concesionario</TITLE>
  </HEAD>
  <BODY>
   <center><h1> Modificar veh&iacute;culo </h1></center>

<?php
   // me conecto a la BD
   include ('conexion.php');

   // Obtengo las variables
   $Modelo = $ REQUEST["id"];

    $consulta = "SELECT * FROM vehiculos WHERE
Modelo='".$Modelo."';";
    $res=mysql query($consulta,$conexion) or die("consulta
incorrecta");
    $fila = mysql fetch array($res);
?>

  <form method="get" action="modificar_guardar.php">
        <table border="1" align="center" cellpadding="3"
cellspacing="0">

          <tr>
                <td><div align="right">Modelo</div></td>
                <td><input name="Modelo" type="text" disabled
value="<?php echo $fila['Modelo'];?>"> </td>
```

```
          </tr>
          <tr>
                <td><div align="right">Motor</div></td>
                <td><input name="Motor" type="text" value="<?php
echo $fila['Motor'];?>"> </td>
          </tr>
          <tr>
                <td><div align="right">Color</div></td>
                <td><input name="Color" type="text" value="<?php
echo $fila['Color'];?>"> </td>
          </tr>
          <tr>
                <td><div align="right">Combustible</div></td>
                <td><input name="Combustible" type="text"
value="<?php echo $fila['Combustible'];?>"> </td>
                </td>
          </tr>
          <tr>
                <td><div align="right">Precio</div></td>
                <td><input name="Precio" type="text" value="<?php
echo $fila['Precio'];?>"> </td>
          </tr>
          <tr>
                <td><div align="right">Marca</div></td>
                <td><input name="Marca" type="text" value="<?php
echo $fila['Marca'];?>"> </td>
          </tr>
          <tr>
                <td colspan=2>
               <div align="center">
                   <input type="submit" name="Submit"
value="Enviar" />
                  </div>
             </td>
          </tr>
        </table>
  </form>
</BODY>
```

modificar_guardar.php

```
<HTML>
  <HEAD>
   <TITLE>Web del concesionario</TITLE>
  </HEAD>
  <BODY>

<?php
include ('conexion.php');

   // Obtengo las variables
   $Modelo =$_REQUEST["Modelo"];
   $Motor = $_REQUEST["Motor"];
   $Color = $_REQUEST["Color"];
   $Combustible = $_REQUEST["Combustible"];
   $Precio = $_REQUEST["Precio"];
   $Marca = $_REQUEST["Marca"];
```

```
    $consulta = "UPDATE vehiculos SET Motor='$Motor',
Color='$Color', Combustible='$Combustible', Precio='$Precio',
Marca='$$
    $res=mysql query($consulta,$conexion) or die("consulta
incorrecta");

    header("Location:index.php");

?>
```

4.10.6 Formulario para eliminar

Este formulario es bastante sencillo, ya que obtiene el modelo del coche (que en nuestro caso es el campo clave), genera y ejecuta la consulta para eliminar el vehículo y vuelve a la página principal.

eliminar.php

```
<HTML>
  <HEAD>
   <TITLE>Web del concesionario</TITLE>
  </HEAD>
  <BODY>

<?php
   // me conecto a la BD
   include ('conexion.php');

   //obtengo el modelo
   $Modelo = $_REQUEST["id"];

   // Creo y construyo la consulta
   $consulta = "DELETE FROM vehiculos WHERE Modelo='".$Modelo."';";
   $res=mysql_query($consulta,$conexion) or die("consulta
incorrecta");

   header("Location:index.php");
?>
```

Capítulo 5

GESTORES DE CONTENIDO (CMS)

5.1 INTRODUCCIÓN

Hace unos años la puesta en marcha de un sitio Web estaba enfocada más a la creación del mismo que a su mantenimiento. En estos sitios, para modificar una página web era necesario conocer las diferentes tecnologías web (p.e. HTML, javascript). Con la evolución de la tecnología web esta tendencia ha ido cambiando y se ha ido separando el contenido de la Web de su diseño. El diseño lo realiza el diseñador Web y el contenido lo gestiona cualquier usuario a través de diferentes formularios que permiten almacenar la información en bases de datos. Estas tendencias han ido evolucionando con el tiempo hasta dar lugar a lo que hoy conocemos como sistemas de gestión de contenido (CMS, Content Management Systems).

Los **gestores de contenido** permiten crear, mantener y gestionar sitios Web con suma facilidad, consiguiendo que los administradores puedan centrarse en tareas más creativas. El entorno proporcionado permite actualizar, mantener y desarrollar la funcionalidad de los sitios por parte de **múltiples usuarios**. Aunque la mayoría permiten gestionar contenido Web (por lo que en ocasiones son denominados *Web Content Management o WCM*), su aplicación no sólo se limita en este campo.

La **funcionalidad de los sistemas de gestión de contenido normalmente incluye cuatro tareas fundamentales**:

- **Creación de contenido**. Un CMS proporciona conjuntos de herramientas que permiten desarrollar contenido en páginas Web para cualquier tipo de usuarios independientemente del conocimiento que tengan. Para ello se utilizan editores de texto del tipo *WYSIWYG (What You See Is What You Get) WYSIWYG* en los que el usuario al mismo tiempo que edita ve los resultados.
- **Gestión de contenido**. Disponen de una base de datos en la que son almacenados todos los documentos creados además de los referentes a la propia Web, datos y preferencia de usuarios, etc.

La estructura de la Web es creada de forma jerárquica y facilitando el **ciclo de trabajo** sobre los contenidos (responsables, editores, autores, usuarios…). En algunos CMS, como los *blogs*, una sola persona hace el papel de administrador, creador y editor.

- **Publicación**. Permite gestionar la publicación de páginas web en el sitio. En los sitios web **existe separación entre diseño y contenido**, algo que es muy importante, ya que podemos modificar el aspecto visual del sitio sin que ello afecte a la información del mismo, permitiendo que cada persona centre sus esfuerzos en su trabajo asignado (administradores en el diseño, autores en el contenido, etc.).
- **Presentación**. Los CMS permiten gestionar de forma automática el acceso a la Web. Por ejemplo, adaptando su presentación a preferencias de usuarios o dando soporte a diferentes navegadores, idiomas, etc. Otros muchos aspectos de la presentación del sitio también son gestionados por este tipo de herramientas, como los menús de navegación o la jerarquía de la página. Por ejemplo, si añadimos un módulo con foros, puede integrar de forma automática en la página principal los últimos mensajes publicados en el mismo.

Los gestores de contenidos presentan una gran flexibiidad y estabilidad, por lo que son de gran utilidad para **sitios en continuo crecimiento, actualización, y dinámicos**. Aparte de lo comentado hasta ahora, también debemos tener en cuenta las siguientes **características**:

- **Inclusión de nuevas funcionalidades**. Gracias a su estructura modular, añadir una nueva funcionalidad al sitio Web es tan sencillo como incluir un módulo con la misma.
- **Mantenimiento de un gran número de páginas**. Permite distribuir los trabajos de creación, edición y mantenimiento mediante el uso de permisos y privilegios de acceso a las diferentes áreas.
- **Reutilización de componentes**.
- **Páginas dinámicas e interactivas**.
- **Aspecto de la Web flexible**. Gracias a la separación entre contenido y presentación, utilizando por ejemplo *CSS* (*Cascading Style Sheets*).
- **Web consistente**. Las páginas son organizadas de forma lógica, haciendo uso de patrones comunes en el diseño de su estructura y estilo.
- **Acceso controlado**. Permite gestionar de forma eficiente los diferentes permisos para acceder a la web, para usuarios y grupos de usuarios.

Existe un **gran número de gestores de contenido disponibles** en la actualidad, lo que permite prácticamente a cualquier nivel elegir y poner en marcha uno que se ajuste perfectamente a nuestras necesidades y objetivos. Normalmente, entre los **criterios a tener en cuenta para la selección de un CMS** u otro tenemos: los objetivos del sitio Web, el tipo de destinatarios, la licencia *open source* o privativo, el grado de desarrollo del mismo, el soporte disponible, su posición en el mercado, la usabilidad y accesibilidad, etc.

A continuación presentamos una clasificación de los gestores de contenido, atendiendo a diversos criterios:

- **En función del lenguaje de programación utilizado en su desarrollo.** Para el desarrollo se CMS se utiliza una gran variedad de tecnologías web entre las que destacan: ASP, Java, PHP, Ruby On Rails, Python, Perl, Ajax, etc.
- **En función del tipo de licencia:**
  - **Open Source**. Al igual que otra aplicación open source las ventajas más importantes son: es gratuito, disponemos del código fuente, podemos mejorarlo, adaptarlo a nuestras necesidades, etc.

    El principal problema de este tipo de soluciones es la escasa documentación disponible (en módulos desarrollados por terceros es aún más patente). En la figura 5-1 y en la tabla 5-1 podemos ver los gestores de contenido *open source* más utilizados actualmente.

  - **Privativos**. Es una aplicación desarrollada para un fin muy específico. En ocasiones se consideran más estables debido a que son desarrolladas por un mismo grupo de desarrolladores (a pesar de esto, las herramientas *open source* también se encuentran coordinadas de manera muy eficiente).
- **Dependiendo del tipo de uso o funcionalidad que aportan.**
  - CMS **de uso general (CMS – Portales Web)**. Permiten la creación de portales Web dinámicos y suelen incluir alguna funcionalidad específica.
  - CMS **de uso específico**. Entre las diferentes categorías de gestores de contenido destacan las siguientes:
    - **Blogs o bitácoras**. Permiten la creación de páginas personales para la publicación periódica de contenidos. Su uso se encuentra muy extendido, sea cual sea el objetivo del *blog* o el tipo de usuario del mismo.
    - **Foros**. En los foros es posible la creación de debates, temas, *hilos*… para compartir opiniones entre usuarios con unos mismos intereses. Al igual que los *blogs* disponen de un gran uso, pudiendo encontrar prácticamente un foro de cualquier temática imaginable.
    - **Wikis**. Este tipo de plataformas están pensadas para el desarrollo colaborativo de contenidos y conocimiento donde, sin duda alguna, el ejemplo más conocido es Wikipedia.
    - **e-learning.** Son plataformas para contenidos de enseñanza on-line o *sistemas de gestión del aprendizaje* (*Learning Management Systems* o *LMS*). Logran facilitar la interacción entre profesorado y alumnado, gestionar contenidos académicos, seguir y evaluar a los estudiantes, etc. El gestor de contenido más importante de *e*-learning es Moodle.
    - **e-commerce**. Se trata de plataformas que incluyen gestión de usuarios, catálogo, compras, gestión de pagos, etc.
    - **Groupware (Software colaborativo)**. Permite integrar el trabajo en un único proyecto llevado a cabo por un número de usuarios concurrentes. Incluye herramientas de comunicación y de

conferencia (chat, vídeo, correo), aunque quizás lo más importante sea el soporte a la gestión de proyectos (documentación, control de versiones…), los sistemas de control del flujo de la actividad y de gestión del conocimiento.

- **Publicaciones digitales**, **difusión de contenido multimedia (Galerías de imágenes…)**.

Figura 5-1. Gestores de contenido open source

En la tabla 5-1 podemos ver los gestores de contenidos open source más importantes.

Tabla 5-1. Gestores de contenido *open source* más importantes

Nombre	URL
Blogs o bitácoras	
WordPress	http://wordpress.org/
Nucleus CMS	http://nucleuscms.org/
Textpattern	http://textpattern.com/
Serendipity	http://www.s9y.org/
CMS – Portales Web	
Plone	http://plone.org/
Jaws	http://www.jaws-project.com/
Joomla	http://www.joomlaspanish.org/
Mambo	http://mambo-foundation.org/
Drupal	http://drupal.org.es/
PHP-Nuke	http://phpnuke.org/
Website Baker	http://www.websitebaker2.org/en/home.php
Kryn.cms	http://www.kryn.org/
Limny	http://www.limny.org/
Alfresco	http://www.alfresco.com/es/
ez Publish	http://ez.no/
Mantis	http://www.mantis.biz/
Redmine	http://www.redmine.org/
e-commerce	
Zen Cart	http://www.zen-cart.com/
Open cart	http://www.opencart.com/
osCommerce	http://www.oscommerce.com/

e-learning	
Moodle	http://moodle.org/
Blackboard	http://www.blackboard.com/
eFront	http://www.efrontlearning.net/
ATutor	http://www.atutor.ca/
Dokeos	http://www.dokeos.com/es
Foros	
SMF	http://www.simplemachines.org/
phpBB	http://www.phpbb-es.com/
PunBB	http://punbb.informer.com/
Vanilla	http://vanillaforums.org/choose
Galerías de imágenes	
Gallery	http://www.gallerycms.com/
Coppermine	http://coppermine-gallery.net/
InterPhoto Image Gallery	http://www.weensoft.com/
ZenPhoto	http://www.zenphoto.org/
Groupware	
Subversion	http://subversion.apache.org/
Webcollab	http://webcollab.sourceforge.net/
Collabtive	http://collabtive.o-dyn.de/
Tine	http://www.tine20.org/
eGroupware	http://www.egroupware.org/
Wiki	
MediaWiki	http://www.mediawiki.org/wiki/MediaWiki/es
WikkaWiki	http://wikkawiki.org/HomePage
DokuWiki	http://www.dokuwiki.org/dokuwiki
PmWiki	http://www.pmwiki.org/
Enano CMS	http://enanocms.org/index
Trac	http://trac.edgewall.org/

Las líneas de trabajo con gestores de contenido en la actualidad y de cara al futuro se encuentran centradas en la incorporación de estándares que permitan una mejor integración de los componentes (p.e. *CSS, XML, XHTML, WAI, RSS*), facilitar el aprendizaje, mejorar la calidad y estabilidad, así como homogeneizar las interfaces de programación de los mismos.

Sin duda alguna, el desarrollo y estandarización de los gestores de contenido ha ido de la mano de la evolución de las herramientas base que le dan soporte y permiten su funcionalidad: los servidores de aplicaciones.

Los servidores de aplicaciones están compuestos por el sistema operativo, servidor web, servidor de base de datos y el lenguaje de programación que se utiliza para desarrollar el gestor de contenido. Por ejemplo, el servidor de aplicaciones más conocido es LAMP (**L**inux, **A**pache, **M**ySQL y **P**HP).

A continuación vamos a aprender a poner en marcha un servidor de aplicaciones LAMP, y posteriormente aprenderemos a instalar y utilizar los gestores de contenido más utilizados.

Nota

Para instalar los gestores de contenido se recomienda que deshabilites selinux.

5.2 JOOMLA!

Joomla! (http://www.joomlaspanish.org/) es un completo gestor de contenidos que facilita enormemente la edición del contenido, interfaz, funcionalidad… de nuestro sitio web. Joomla! está desarrollado en PHP y dispone de licencia GPL. Joomla! puede ser utilizado para la creación de sitios para prácticamente cualquier fin: *blogs,* creación de comunidades, *intranets*, foros y sitios multimedia, comercio electrónico… o incluso es posible hacer todo a la vez.

5.2.1 Requisitos previos

Antes de descargar el software Joomla! debemos asegurarnos de cumplir los siguientes requisitos, que consisten en la instalación de los componentes LAMP vistos antes. Para la versión de Joomla! que vamos a instalar (1.5.17) son:

- PHP 4.2.x o superior.
- MySQL 3.23.x o superior.
- Apache 1.13.19 o superior.
- PHP debe tener instalado el soporte para MySQL, XML y Zlib.

Comprobamos que efectivamente disponemos de este soporte accediendo a través del navegador a la página web *phpinfo.php* utilizada anteriormente (véase la figura 5-2).

A continuación creamos la base de datos de *joomla* y establecemos los permisos de acceso al *usuario joomla@localhost* de la siguiente forma:

```
# mysql –u root -p
mysql> create database joomla;
mysql> quit;
```

Establecemos todos los privilegios al usuario:

```
mysql> GRANT ALL PRIVILEGES ON joomla.* TO joomla@localhost
IDENTIFIED BY 'contrasena';
mysql> flush privileges;
mysql> quit;
```

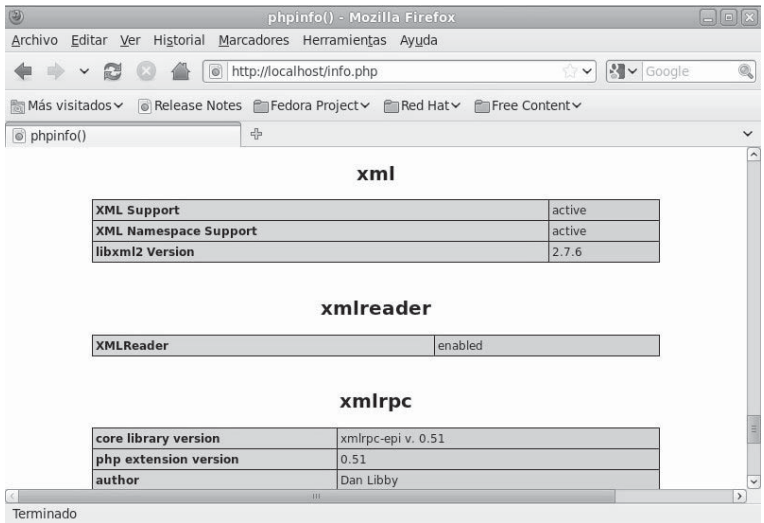


Figura 5-2. Comprobando el módulo xml con phpinfo()

5.2.2 Instalación

Para comenzar con la instalación de Joomla! entramos en el directorio de nuestro sitio web */var/www/html* y descargamos el paquete Joomla! haciendo uso de la utilidad *wget* (si queremos instalar Joomla! en otro directorio que no sea el accesible por defecto podemos por ejemplo crear un subdirectorio en él y seguir los mismos pasos).

```
# cd /var/www/html
# wget
http://joomlacode.org/gf/download/frsrelease/12205/49808/Joomla_1.5.
17-Spanish-pack_completo.tar.gz
```

Descomprimimos el paquete ejecutando:

```
# tar xzvf  Joomla_1.5.17-Spanish-pack_completo.tar.gz
```

A continuación escribimos en el navegador la dirección http://localhost/ y realizamos la instalación a través del asistente web:

- Seleccionamos el idioma que queremos utilizar durante la instalación (véase la figura 5-3).

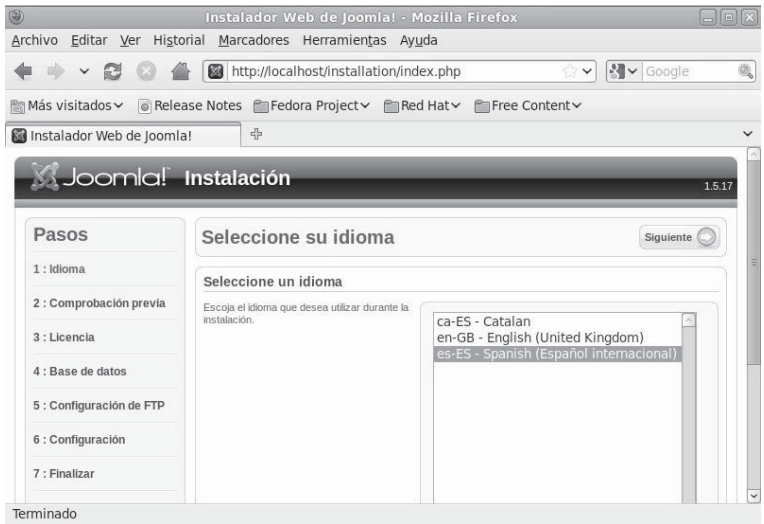


Figura 5-3. Joomla! – Instalación – Selección de idioma

- El sistema realiza una comprobación previa para ver si se cumplen los requisitos necesarios para instalar Joomla!.

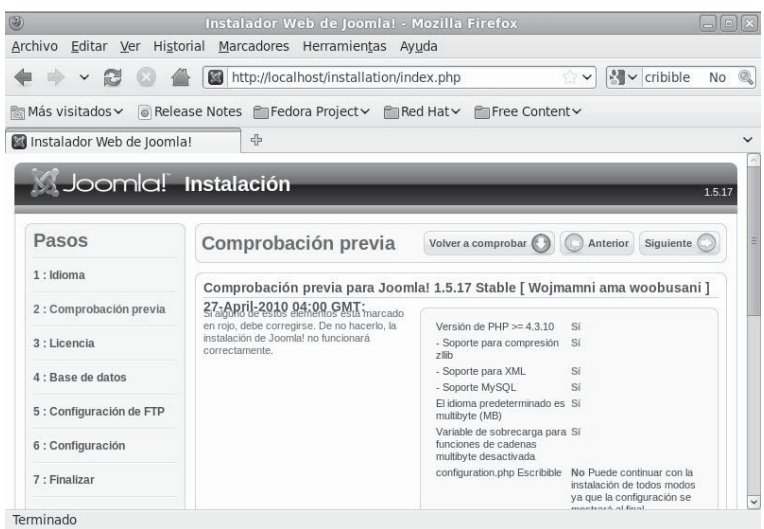


Figura 5-4. Joomla! – Instalación – Comprobación previa

- Una vez que cumplimos los requisitos aceptamos los términos de la licencia GPL.
- A continuación configuramos el acceso a la base de datos MySQL. Para ello debemos especificar el tipo de base de datos (MySQL), nombre de usuario

(joomla), contraseña (‘contrasena’), nombre de la base de datos (joomla) y en configuración avanzada podemos administrar la base de datos actual.

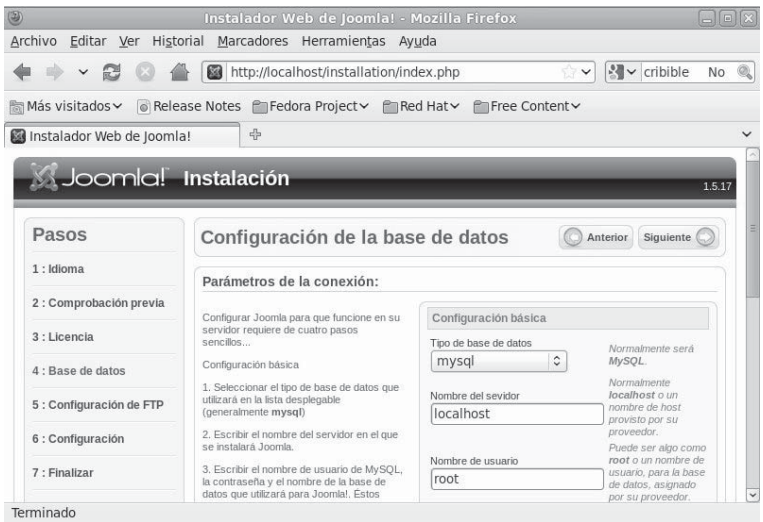


Figura 5-5. Joomla! – Instalación – Configuración de la base de datos

- Configurar, si es necesario, el acceso mediante FTP a los directorios Joomla! en nuestro servidor web.

Figura 5-6. Joomla! – Instalación – Configuración de FTP

- Especificamos el nombre del sitio web, el correo electrónico y la contraseña del usuario *admin*. Más adelante podemos elegir si instalar datos de ejemplo y restaurar o migrar el contenido desde un sitio web anterior.

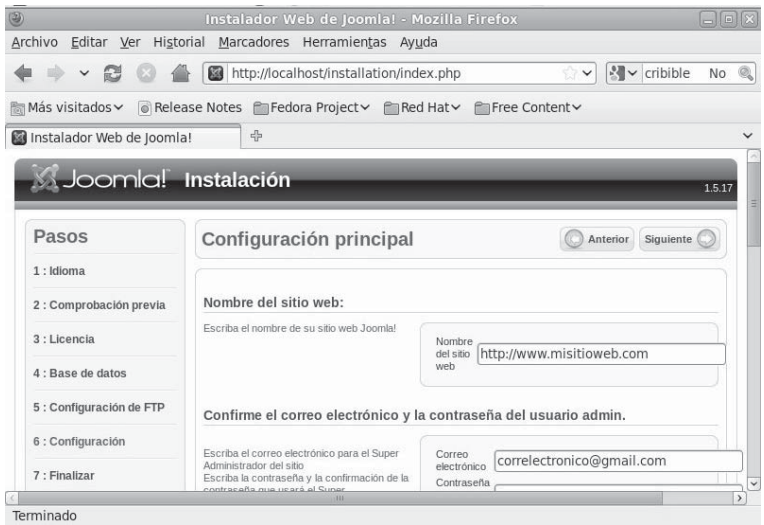


Figura 5-7. Joomla! – Instalación – Configuración principal

- Y finalizamos la instalación.

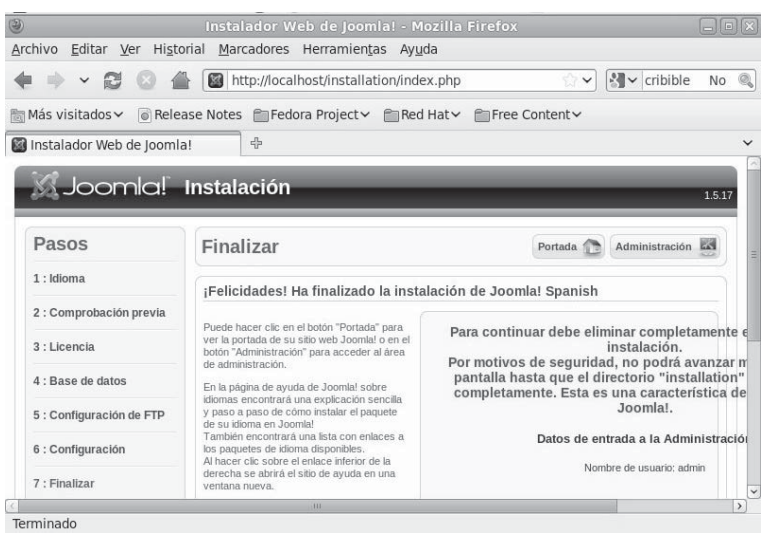


Figura 5-8. Joomla! – Instalación – Finalizar

Antes de acceder a la portada de nuestro nuevo sitio web o a la administración del mismo mediante los accesos que vemos debemos eliminar el directorio *installation* y todo su contenido por motivos de seguridad, ubicado en el directorio */var/www/html*.

```
# rm –dfr /var/www/html/installation/
```

Una vez hecho esto podemos ver que se nos facilitan los datos de acceso y el código de la configuración que debemos copiar en el directorio del sitio web, ya que en principio no tenía permisos de escritura para hacerlo de forma automática. Creamos por tanto este fichero y lo editamos copiando el contenido facilitado:

```
# gedit /var/www/html/configuration.php
```

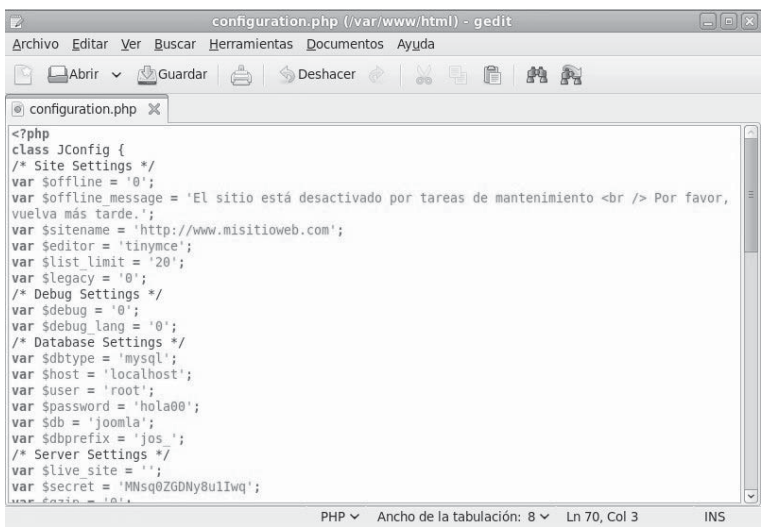


Figura 5-9. Joomla! – configuration.php

Ahora sí podemos acceder a la portada de Joomla! mediante el enlace correspondiente en el último paso de la instalación o a través de la dirección http://localhost/ (o la del directorio en el que haya sido instalado):

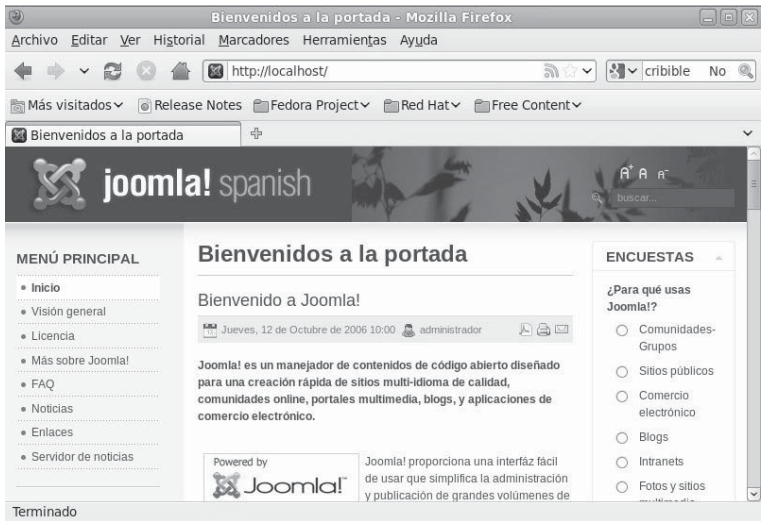


Figura 5-10. Joomla! – Portada

Si queremos acceder a la zona administrativa del portal Joomla! escribimos http://localhost/administrator, con el usuario *admin* y la contraseña que hemos indicado durante al proceso de instalación.

Figura 5-11. Joomla! – Instalación – Acceso a la administración

Desde esta zona ya podemos comenzar a configurar cualquier aspecto de nuestro sitio web Joomla!: añadir un artículo, gestionar los artículos, la página principal, las secciones que se muestran, qué menús están disponibles, etc.

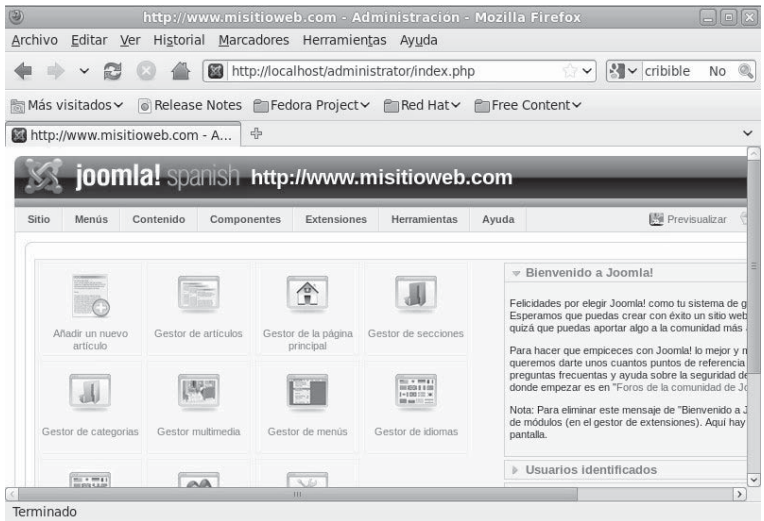


Figura 5-12. Joomla! – Administración

Más información

Manual completo de Joomla!
http://ayuda.joomlaspanish.org/ayuda-joomla/

5.2.3 Utilización

En este apartado vamos a comentar los rasgos principales de la funcionalidad aportada por Joomla! repasando cuáles son las operaciones básicas que podemos realizar una vez lo hemos instalado. Las podemos dividir desde dos puntos de vista: administración desde el *backend* o *panel de administración* y administración desde el *frontend* o parte pública de nuestro sitio.

5.2.3.1 EL PANEL DE CONTROL (BACKEND)

Para acceder a la *zona de administración* de nuestro sitio Web (es decir, la parte *backend*) el primer paso que debemos hacer es acceder a la *zona de administración*. Para ello accedemos con el navegador a la dirección http://nombre del sitio/administrator/ y escribimos *admin* como usuario y la contraseña que hayamos configurado durante el proceso de instalación (véase la figura 5-13).

Figura 5-13. Joomla! – Acceso al panel de control

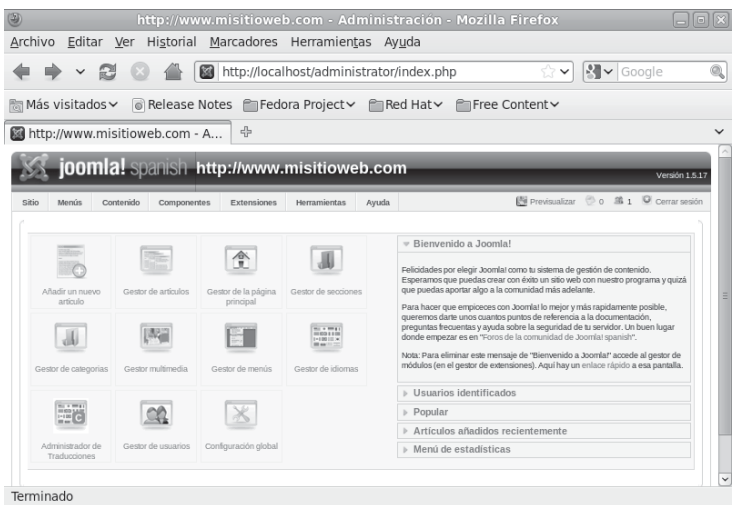


Figura 5-14. Joomla! – Vista general del panel de control

Tal y como se muestra en la figura 5-14, una vez dentro podemos observar diferentes elementos que nos permiten configurar nuestro sitio al completo, donde las opciones más importantes son:

- La **barra de menús.** Contiene menús desplegables con las diferentes operaciones a realizar, elementos de la página que podemos administrar, etc., agrupados en categorías. Siempre la encontramos de manera fija en la parte superior de la ventana, independientemente de la configuración que estamos realizando en el momento.

Figura 5-15. Joomla! – Barra de menús del panel de control

Las categorías mostradas son las siguientes:

- **Sitio**. Permite establecer la configuración de aspectos generales para el sitio Web. Podemos encontrar en él vínculos para acceder de nuevo al *Panel de Control* (ventana inicial cuando entramos a la zona de administración), administrar los usuarios del sitio y sus privilegios (*Gestor de usuarios*), gestionar los elementos multimedia alojados en el sitio (*Gestor multimedia*), ver y cambiar la *configuración global* así como *cerrar sesión*.

  En la figura 5-16 podemos ver el aspecto de la ventana *Configuración global*, donde podemos configurar el nombre del sitio, el editor de texto por defecto, número de elementos por página, longitud de las noticias, la descripción del sitio, etc.

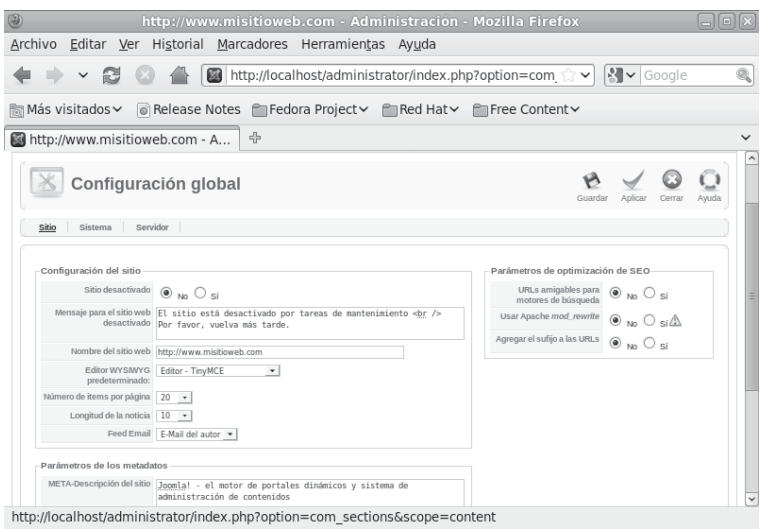


Figura 5-16. Joomla! – Configuración global

- **Menús.** Permite gestionar los menús que forman parte de nuestro sitio. Para ello incluye un completo *Gestor de menús* (figura 5-17) desde el cual podemos ver los actuales, editar los elementos que contienen, crear nuevos, eliminarlos, etc.

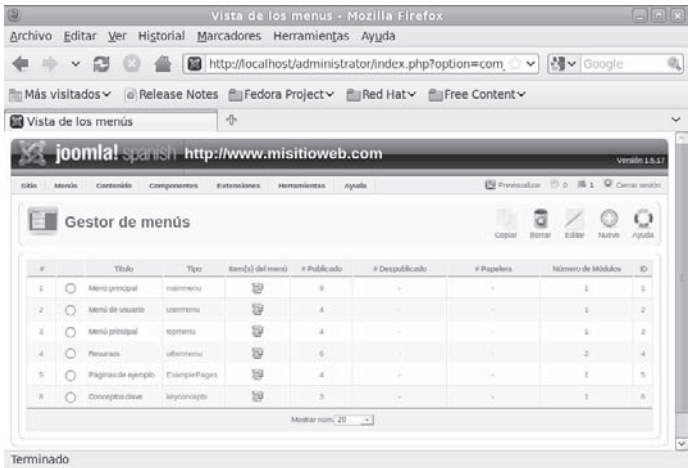


Figura 5-17. Joomla! – Gestor de menús

Además tenemos disponible una **Papelera de menús** que almacena menús que hemos dejado de publicar y que queremos "marcar" como eliminados, así como accesos directos para la manipulación de todos los menús en uso para el sitio y los diferentes elementos que contienen.

- **Contenido.** Permite gestionar los contenidos publicados y por publicar en nuestro sitio, es decir, los **artículos, secciones y categorías** que lo conforman y estructuran. En la figura 5-18 podemos ver el *Gestor de artículos* donde podemos ver todos los artículos creados, archivarlos, ver en qué sección están ubicados y a qué categoría pertenecen, editarlos, cambiar las preferencias para los mismos, etc.

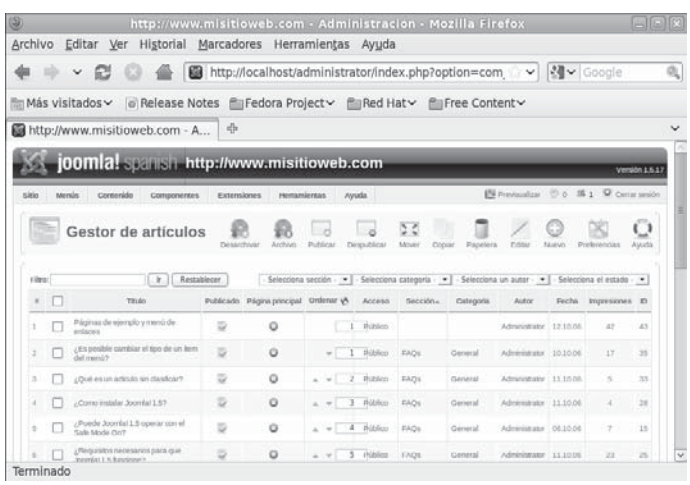


Figura 5-18. Joomla! – Gestor de artículos

Además de la gestión de artículos podemos administrar desde este menú las secciones y categorías que conforman nuestra Web, así como gestionar específicamente la página principal del sitio y los elementos que aparecen en ella mediante el vínculo *Gestionar la página de inicio*.

- **Componentes.** Desde este menú podemos administrar los diferentes *componentes* que han sido integrados en nuestro sitio. Los *componentes* son elementos funcionales del núcleo de Joomla! y se muestran en el cuerpo principal de la plantilla del sitio web. Dependiendo del diseño de la plantilla en uso, pueden situarse en el centro de la página web o en cualquier otro lugar determinado. Por ejemplo por defecto aparecen los siguientes *componentes*: *banners* publicitarios, *buscar, contactos, encuestas, enlaces Web, servidor de noticias.*

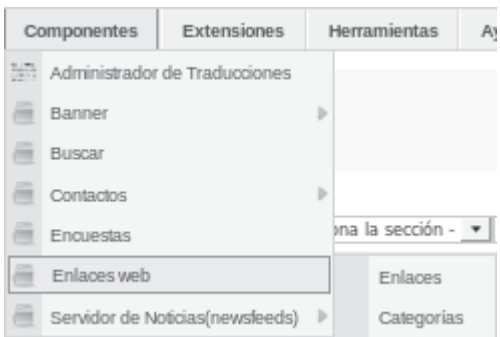


Figura 5-19. Joomla! – Menú componentes

También se incluye el *Administrador de Traducciones*, que nos permite definir el(los) idioma(s) para los clientes.

- **Extensiones.** Permite *instalar y desinstalar* las extensiones para el sitio y gestionar los **módulos, plantillas, idiomas y *plugins*** del mismo. Los módulos se utilizan para mostrar diversa información o características interactivas en el sitio Joomla!. Desde los diferentes gestores disponibles podemos administrar cada uno de estos elementos para nuestro sitio: ver si están habilitados, su posición, en qué páginas aparecen, etc.

  Si pulsamos el enlace *Instalar/Desinstalar* abrimos la ventana *Gestor de extensiones* (véase la figura 5-20) desde la cual podemos instalar nuevas extensiones (esto es, módulos, componentes, *plugins*, plantillas…) haciendo clic en *Instalar* o desinstalarlas si vamos a cada uno de los tipos de extensiones.

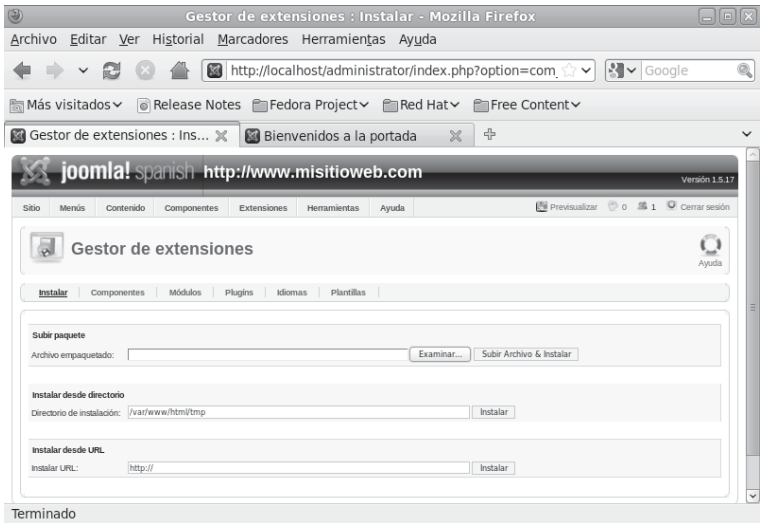


Figura 5-20. Joomla! – Gestor de extensiones

La instalación de los diferentes tipos de extensiones puede ser realizada de tres formas distintas: **subiendo e instalando un paquete, instalando la extensión desde un directorio local del sistema o desde una URL**.

- **Herramientas.** Aquí encontramos accesos directos a **herramientas de comunicación**: leer mensajes recibidos, escribir mensajes, enviar correo masivo a todos los usuarios… También podemos desbloquear de forma global el sitio o **limpiar la caché** del mismo.

- **Ayuda.** En *Ayuda*, además de un enlace a la ayuda general de Joomla!, disponemos del vínculo *Información del sistema*: estado de la configuración PHP, versión de la base de datos, versión de Joomla!, tipo de servidor Web, permisos de acceso para los diferentes directorios de trabajo… Además haciendo clic en *Archivo de configuración* podemos ver el contenido de éste vía Web.

  Al final de los diferentes menús de la barra tenemos un acceso directo para la previsualización del sitio (*Previsualizar*), para cerrar la sesión de administrador o para leer la bandeja de entrada de mensajes. Así mismo encontramos disponible un contador con el número de usuarios del sitio.

- **Menú central**. Contiene botones que son accesos directos a los principales y más habitualmente utilizados elementos administrativos y operaciones de los comentados anteriormente para la barra de menús: *Añadir un nuevo artículo, Gestor de artículos, Gestor de la página principal, Gestor de secciones, Gestor de categorías, Gestor multimedia, Gestor de menús, Gestor de idiomas, Administrador de traducciones, Gestor de usuarios y Configuración global.*

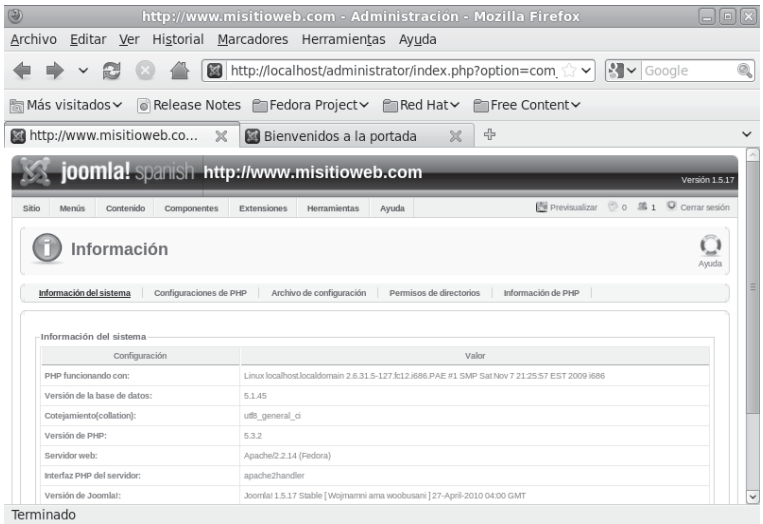


Figura 5-21. Joomla! – Información del sistema

Figura 5-22. Joomla! – Menú central

- **Menú desplegable con la actividad del sitio.** Situado en la parte derecha del *panel de control* muestra diferentes menús entre los cuales podemos navegar para observar algunos aspectos de la actividad reciente del sitio. Por ejemplo, los *Usuarios identificados* en este momento (véase la figura 5-23), cuáles son los artículos más populares del sitio (*Popular*), la información sobre los *Artículos añadidos recientemente*, así como un interesante *Menú de estadísticas* que nos permite ver el número de elementos que contiene cada menú.

Figura 5-23. Joomla! – Actividad reciente del sitio

5.2.3.2 GESTIÓN PÚBLICA DE LOS ARTÍCULOS (FRONTEND)

Es el momento de conocer también a grandes rasgos cómo administrar nuestro sitio Web Joomla! haciendo uso del acceso público disponible para el mismo. Para ello, en la página principal del sitio localizamos el **formulario de acceso** (normalmente situado en la parte izquierda de la misma) e introducimos nuestro nombre de usuario y contraseña.

Tras pulsar *Iniciar sesión* y si no hemos cometido ningún error en la introducción de los datos, podemos apreciar entonces que el aspecto de nuestro sitio Web ha cambiado. En primer lugar, junto al formulario de acceso (que ahora contiene un mensaje de bienvenida y un botón para abandonar la sesión) aparece una sección denominada *Menú de usuario* desde la cual podemos ver y editar nuestro perfil, enviar un artículo al sitio, enviar un enlace, y también permite finalizar la sesión (véase la figura 5-24).

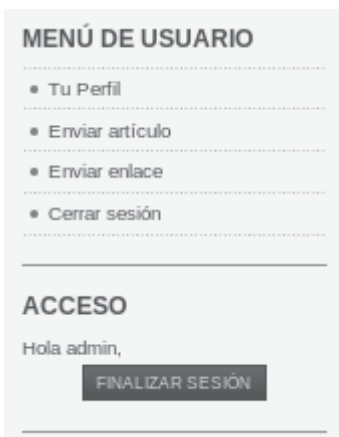


Figura 5-24. Joomla! – Menú de usuario y Acceso

Además, junto al título de los artículos ahora disponemos de un enlace con el icono que nos permite **editar el artículo** así como observar de forma rápida sus propiedades generales: si se encuentra publicado, la difusión del artículo, quién lo ha

publicado y en qué fecha. Haciendo clic en este icono vemos una **versión editable del artículo**, como podemos ver en la figura 5-25.

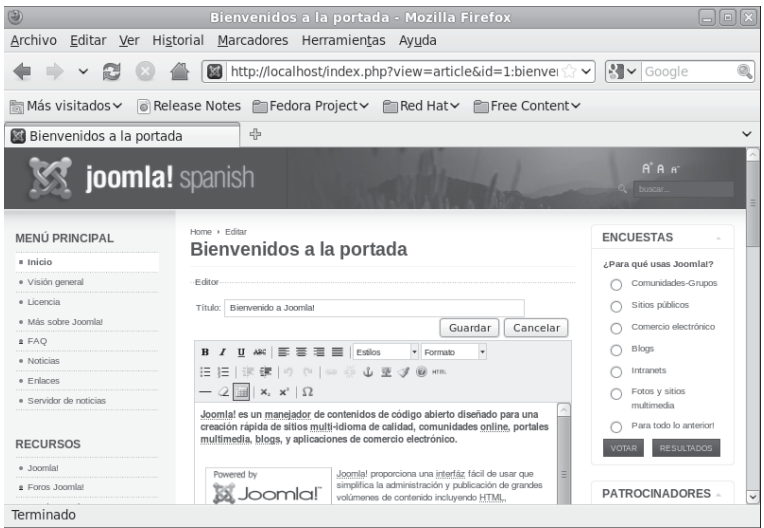


Figura 5-25. Joomla! – Edición de un artículo

Desde esta versión podemos cambiar el título del artículo, el texto del contenido, insertar imágenes, editar el contenido en formato HTML, insertar un botón *Leer más*…; una vez que hayamos finalizado o cada vez que queramos podemos guardar los cambios realizados pulsando el botón *Guardar*. Si una vez que estamos editando el contenido de un artículo pulsamos alguno de los botones *Atrás, Adelante, Cerrar…* del navegador, lo más probable es que el artículo quede bloqueado. Es entonces cuando el administrador debe desbloquearlo accediendo al *Panel de control* de Joomla!.

La parte que aparece por encima de la línea de puntos rojos es la llamada *entradilla* del artículo, es decir, el texto que aparece como introducción al mismo y que capta a los lectores para que lo lean al completo, pulsando en *Leer más*. Con la opción *Paginar* podemos estructurar el artículo en varias páginas, creando para esto los menús correspondientes en la vista pública del mismo (enlaces a las páginas, *página anterior, página siguiente,* etc.). Esta opción es muy importante cuando el contenido del artículo es extenso.

Debajo del editor del contenido del artículo disponemos de diferentes opciones que nos permiten caracterizar la publicación (*Publicando*), que podemos ver en la figura 5-26.

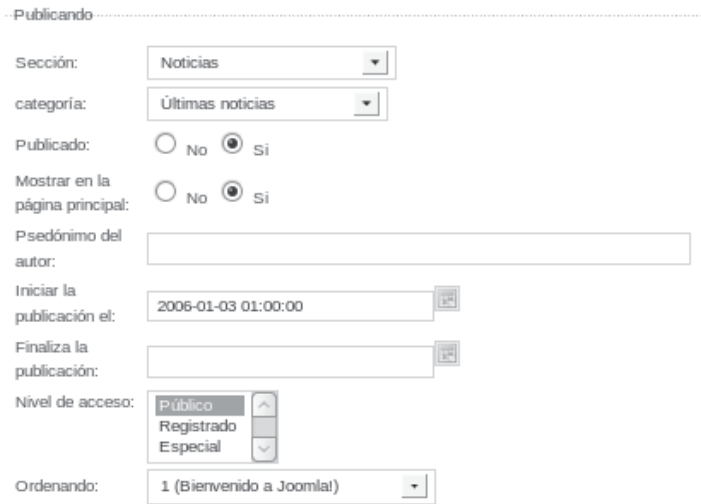


Figura 5-26. Joomla! – Publicando

Elegimos la sección en la que ubicar el artículo y la categoría a la que pertenece, si se encuentra visible en el sitio (*Publicado*), si lo queremos mostrar en la página principal, o podemos escribir un seudónimo para el autor si no queremos mostrar su nombre real. Una opción importante es *Iniciar la publicación el*, que nos permite indicar fecha y hora para que el artículo pase a ser visible de forma automática. De igual forma podemos especificar una fecha en la que el artículo deja de estar disponible. Finalmente podemos indicar el orden para el artículo y el nivel de acceso para el mismo: *Público* (para todos los clientes que accedan al sitio), *Registrado* (sólo disponible para usuarios registrados en el sitio) y *Especial* (por ejemplo, si queremos crear artículos sólo disponibles para usuarios especiales o administradores).

5.3 MOODLE

Moodle (http://moodle.org/) es un *Sistema de Gestión de Cursos de Código Abierto* (o *CMS, Open Source Course Management System*) o también conocido como *Plataforma de Enseñanza Virtual*. De esta forma es sencillo para educadores crear y establecer sitios web de aprendizaje (aulas virtuales) accesibles ininterrumpidamente para el alumnado, con todas las posibilidades que ello ofrece (foros, chats, entrega de actividades, etc.).

Entre las funcionalidades más importantes de Moodle destacan:

- Administración de **profesorado y alumnos**.
- Creación y establecimiento de **aulas virtuales**, con toda la información sobre el curso, los contenidos académicos del mismo y proporcionando medios de comunicación entre profesores y alumnos: **foros, chats, mensajería privada**, etc.
- El **material de un curso** puede ser creado, gestionado y publicado en diferentes formatos.

- **Exámenes y test de evaluación** on-line.
- **Entrega de actividades** con fecha límite y posible envío de avisos al profesor sobre las mismas.
- **Generación de estadísticas** con las acciones realizadas por los alumnos.

5.3.1 Requisitos previos

Antes de comenzar a instalar Moodle debemos contar como es lógico con un servidor web instalado (Apache), así como PHP y los módulos necesarios para su interacción con la base de datos que vayamos a utilizar (MySQL).

Para realizar la instalación es necesario crear una base de datos (p.e. *moodle*):

```
# mysql –u root -p
mysql> create database moodle;
mysql> quit;
```

Y establecer los privilegios sobre la base de datos a un determinado usuario (p.e. moodle):

```
mysql> GRANT ALL PRIVILEGES ON moodle.* TO moodle@localhost
IDENTIFIED BY 'contrasena';
mysql> flush privileges;
mysql> quit;
```

Finalmente creamos un directorio el cual Moodle va a utilizar para almacenar todos los datos que vaya generando. Lo llamamos moodledata y lo ubicamos por ejemplo en /var/www/ (es recomendable que sea en un directorio que no pertenezca al sitio web) y que se limite el acceso por el usuario moodle.

```
# adduser moodle
# mkdir /var/www/moodledata
# chmod 775 /var/www/moodledata –R
# chown moodle:moodle /var/www/moodledata -R
```

Nota

Tenga cuidado con los permisos del directorio *moodledata* ya que en él se encuentran todos los datos del servidor moodle.

5.3.2 Instalación

Para comenzar el proceso de instalación de Moodle descargamos la última versión disponible desde su web, por ejemplo en el enlace: http://download.moodle.org/download.php/stable19/moodle-weekly-19.tgz.

Copiamos el paquete descargado al directorio de nuestro sitio web (por defecto */var/www/html*), lo descomprimimos y eliminamos el paquete original:

```
# cp moodle-weekly-19.tgz /var/www/html
# cd /var/www/html
# tar xvfz moodle-weekly-19.tgz
# rm moodle-weekly-19.tgz
```

De esta forma todos los ficheros de Moodle han sido descomprimidos en el directorio */var/www/html/moodle* y escribiendo en el navegador la dirección http://localhost/moodle/ podemos comenzar con el proceso de instalación:

- Seleccionamos el idioma para la instalación (véase la figura 5-27).

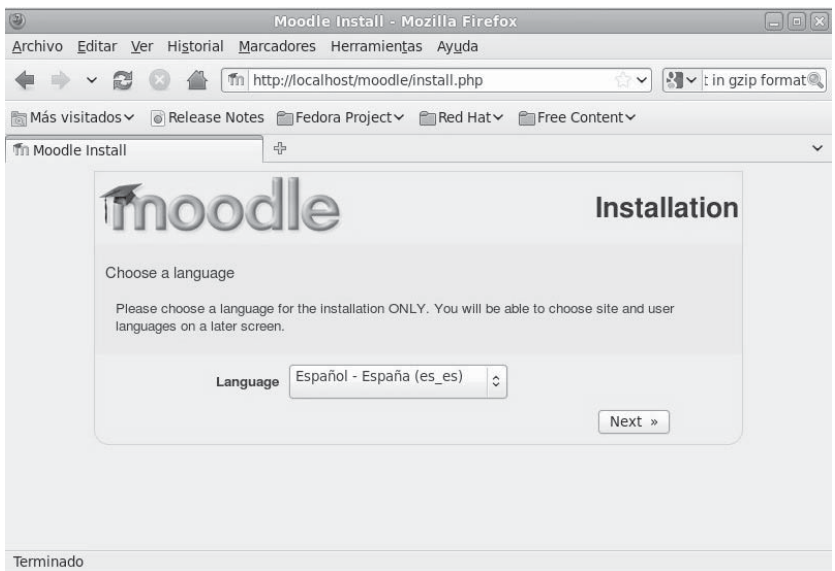


Figura 5-27. Moodle – Instalación – Selección de idioma

- El sistema comprueba que los ajustes PHP en nuestro servidor están configurados correctamente. De lo contrario, es momento para realizar los cambios oportunos.

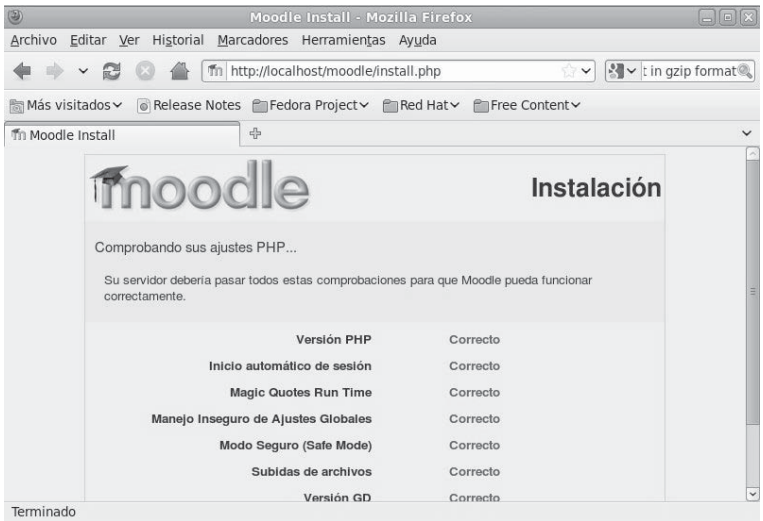


Figura 5-28. Moodle – Instalación – Comprobación de PHP

- Escribimos el DNS del sitio web y el directorio de datos para Moodle (directorio *moodledata* que hemos creado en pasos anteriores):

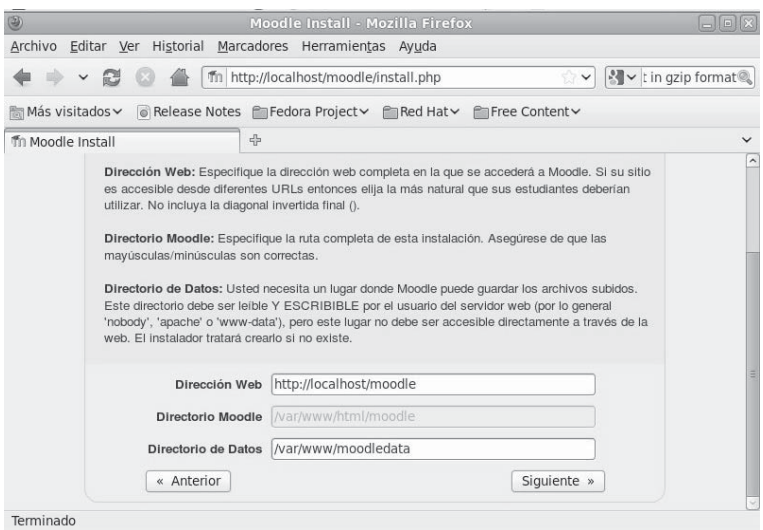


Figura 5-29. Moodle – Instalación – Configuración del sitio

- A continuación debemos configurar el acceso de Moodle. Seleccionamos el tipo de base de datos MySQL, la dirección del servidor, el nombre de la base de datos, y el nombre de usuario y la contraseña de acceso.

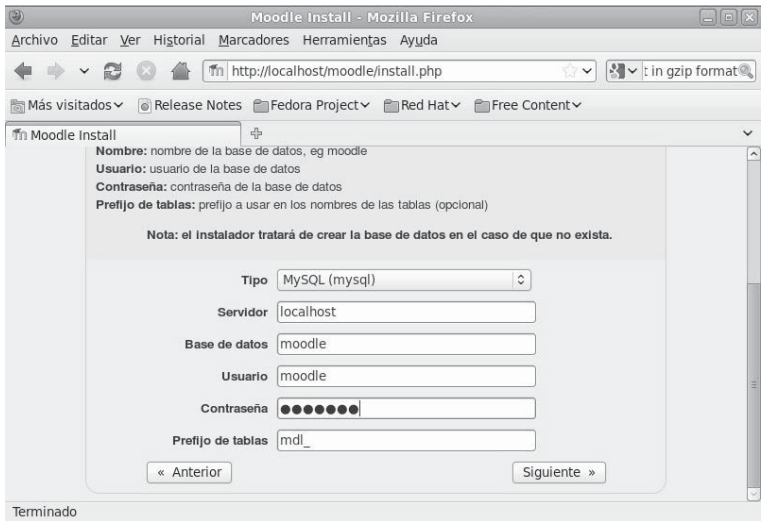


Figura 5-30. Moodle – Instalación – Configuración de la base de datos

- El instalador comprueba el entorno del servidor para Moodle y nos presenta un informe con el estado de todos los requisitos que debemos cumplir. Cuando todo es correcto pulsamos *Siguiente*:

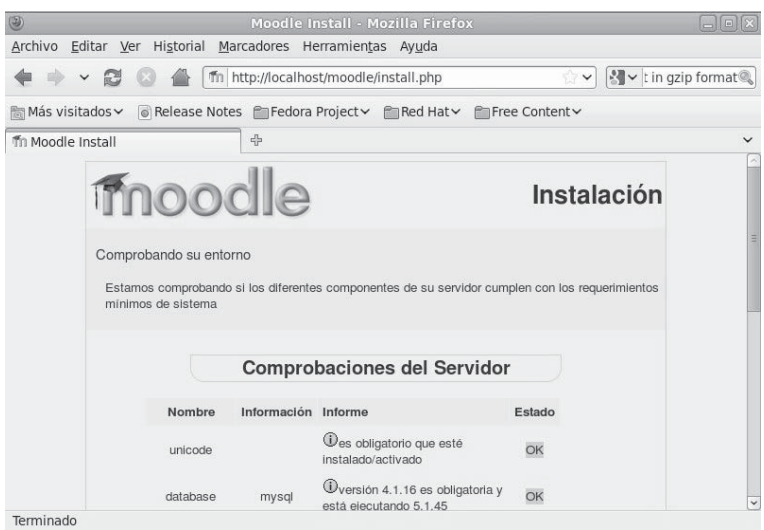


Figura 5-31. Moodle – Instalación – Comprobaciones del servidor

- Descargamos el paquete del idioma *Español* y pulsamos *Siguiente*:

Figura 5-32. Moodle – Instalación – Descarga del paquete de idioma Español

- Al no tener permisos de escritura *Apache* sobre el directorio de instalación no se puede generar automáticamente el fichero de configuración *config.php*. Por lo tanto, debemos crear manualmente el fichero (*nano /var/www/html/moodle/config.php*) y copiar el contenido que se nos muestra en pantalla.

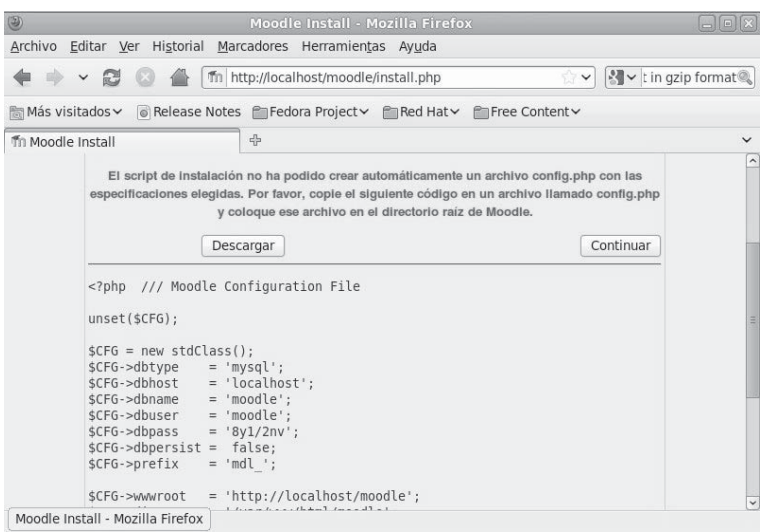


Figura 5-33. Moodle – Instalación – Script config.php

- Aceptamos los términos de la licencia GPL de la aplicación.

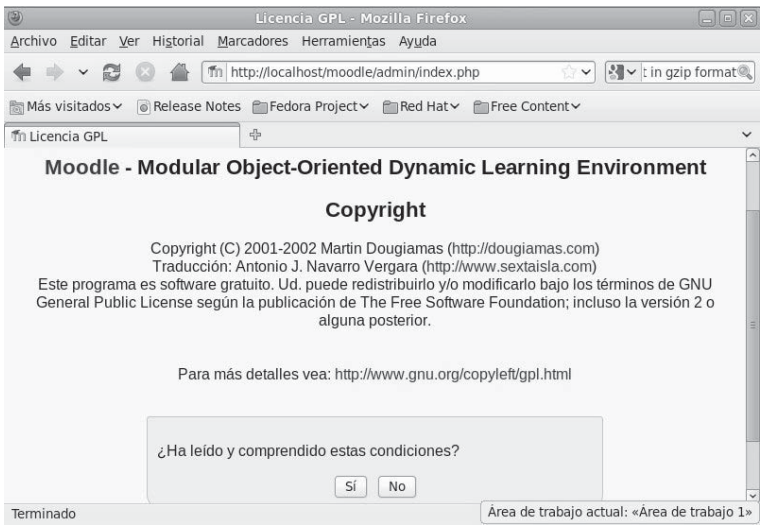


Figura 5-34. Moodle – Instalación – Condiciones

- A continuación el sistema va a iniciar el proceso de creación de la estructura de la base de datos. Esta operación la podemos realizar de manera desatendida o paso por paso. Lo más rápido es realizarla de forma desatendida.

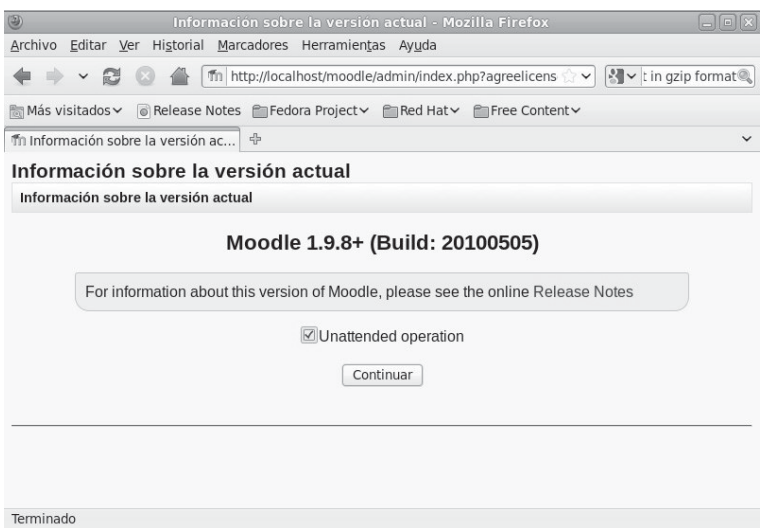


Figura 5-35. Moodle – Instalación – Versión e instalación desatendida

- El instalador procede con la puesta a punto de la base de datos, instalando el módulo de tablas, configurando tablas de *plugins*, tablas de bloque, etc. (véase la figura 5-36).

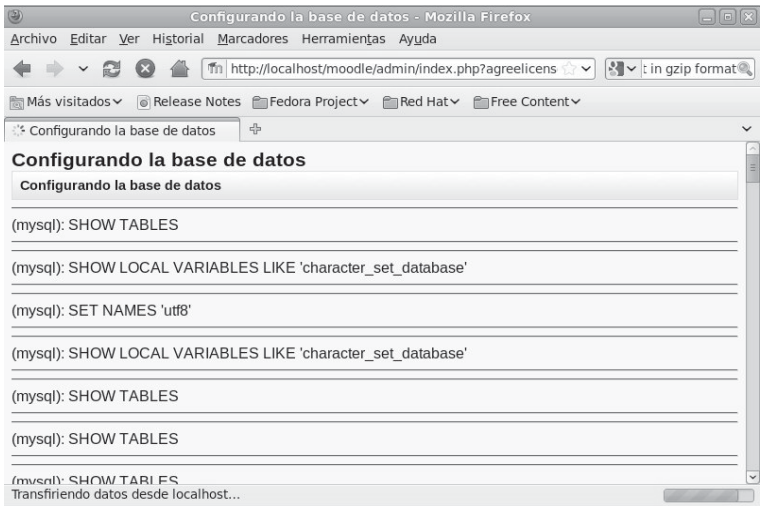


Figura 5-36. Moodle – Instalación – Configurando la base de datos

- Cuando finaliza el proceso de creación de la estructura de la base de datos, debemos establecer los datos de la cuenta del administrador para Moodle (nombre de usuario, contraseña, nombre y apellidos, correo electrónico). Cuando finalizamos de incluir toda la información pulsamos el botón *Actualizar información personal.*

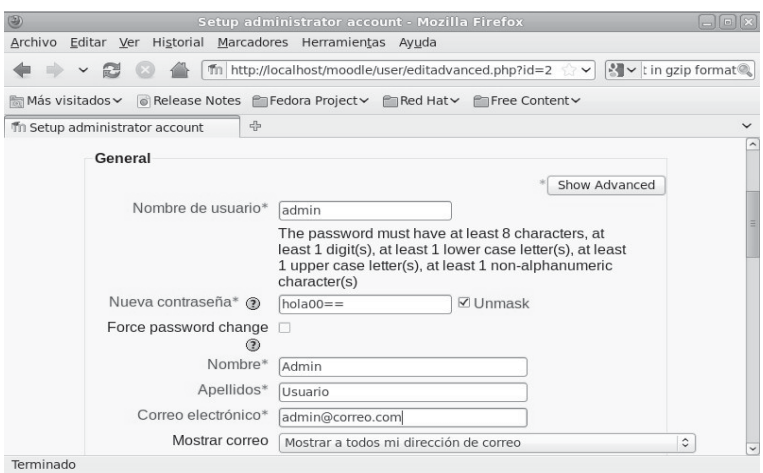


Figura 5-37. Moodle – Instalación – Configuración de la cuenta del administrador

- Establecemos la información sobre el sitio (nombre, descripción, encabezado de la página, etc.). Para finalizar pulsamos el botón *Save Changes*.

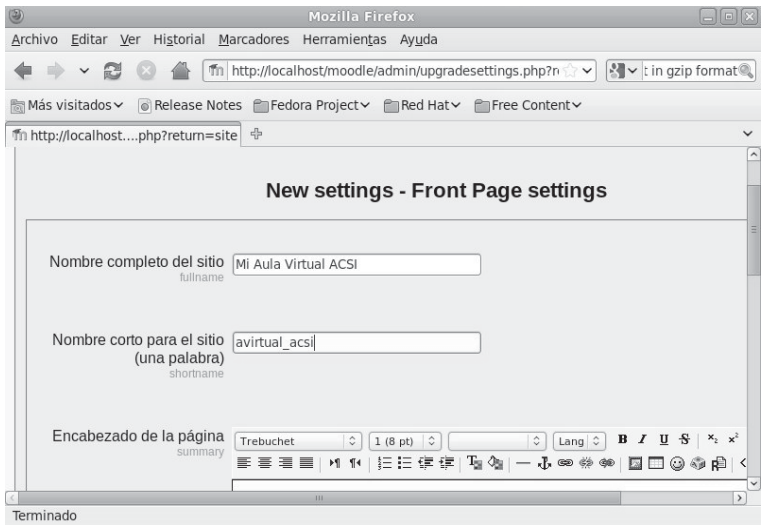


Figura 5-38. Moodle – Instalación – Configuración de la página principal (I)

Ya podemos visitar la página principal del aula virtual, bien haciendo clic en el botón que hay disponible para ello o a través de la dirección http://localhost/moodle/:

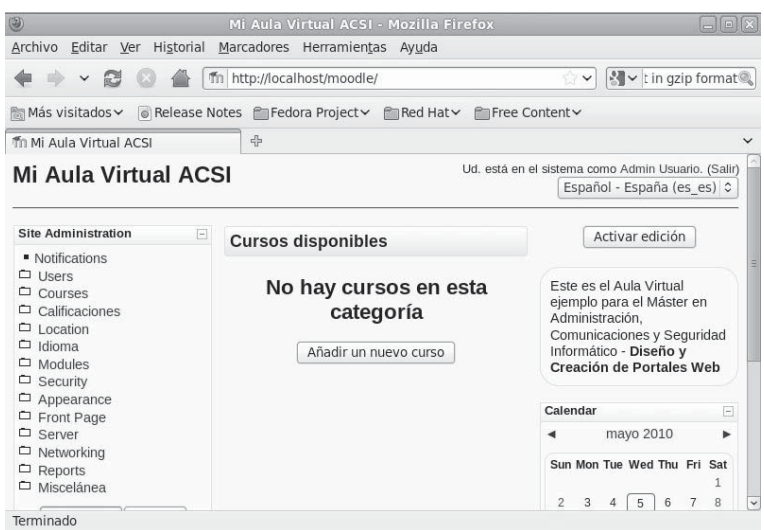


Figura 5-39. Moodle – Página principal del aula virtual

Desde ahora podemos comenzar a crear nuevos cursos, añadir alumnos, seleccionar las características disponibles para los cursos… en definitiva caracterizar al completo el contenido de Moodle.

Más información

Documentación oficial de Moodle
http://docs.moodle.org/en/Main_Page

5.3.3 Utilización

Para comenzar a administrar Moodle vamos al navegador Web y escribimos la ruta en la que ha sido instalado, normalmente http://nombre_del_sitio/moodle, o http://localhost/moodle si nos encontramos en el mismo equipo en el que está instalado.

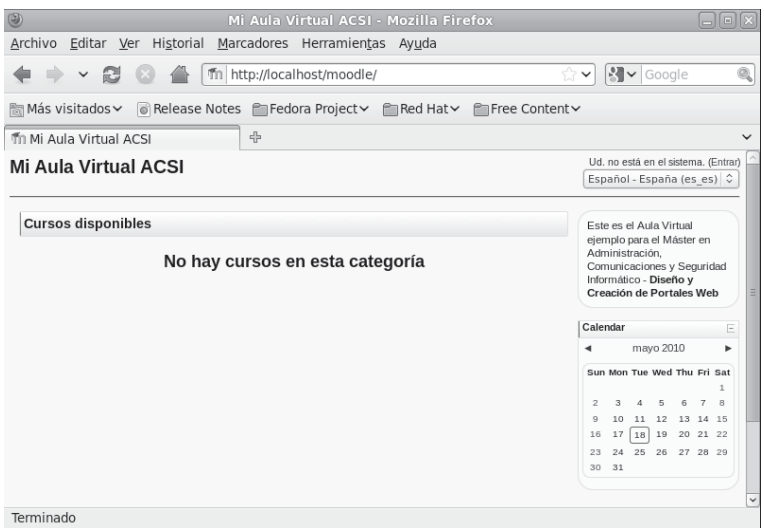


Figura 5-40. Moodle – Inicio

Para acceder al aula virtual como administrador hacemos clic en el enlace *Entrar*, disponible en la esquina superior derecha de la ventana. Es entonces cuando aparece el *Formulario de acceso al sitio* (véase la figura 5-41), en él escribimos el nombre de usuario y la contraseña del administrador que fue especificada durante el proceso de instalación.

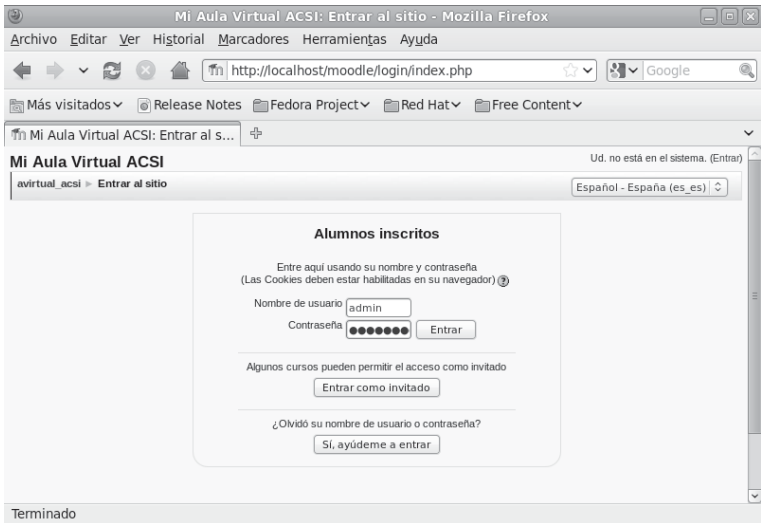


Figura 5-41. Moodle – Formulario de acceso

Una vez autenticados, aparece la página de inicio para el aula virtual Moodle tal y como aparece en la figura 5-42.

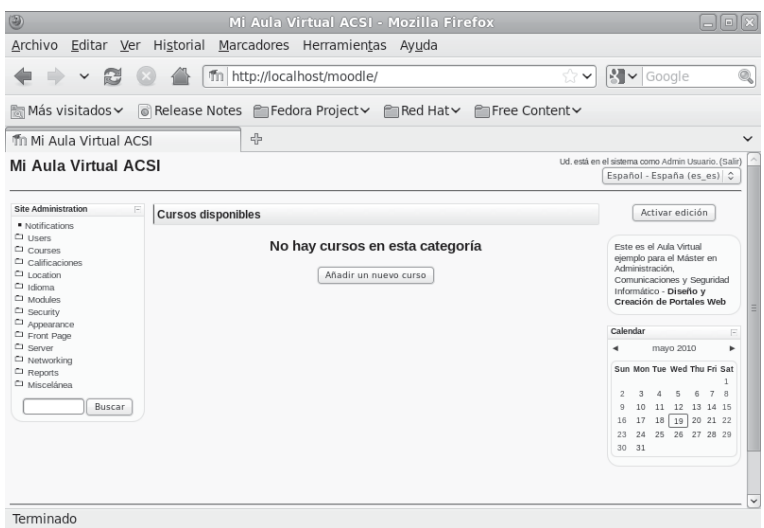


Figura 5-42. Moodle – Inicio del aula virtual

5.3.3.1 ADMINISTRACIÓN DEL SITIO

En la parte derecha de la pantalla inicial podemos ver cómo aparece la descripción introducida en la instalación para el aula virtual así como un *Calendario*. Es posible navegar

por las diferentes fechas del calendario en el que, una vez creadas, pueden mostrarse diferentes eventos como entregas de actividades, fecha de publicación de calificaciones, etc.

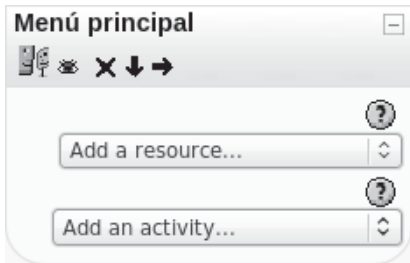


Figura 5-43. Moodle – Iconos para la edición

Si hacemos clic en el botón *Activar edición* podemos mostrar una serie de elementos que nos permiten cambiar la configuración de los diferentes bloques que contiene la página (esto es igual para todas las páginas del aula virtual), incluso ver los bloques de la página que permanecen ocultos. Por ejemplo, en la figura 5-44 podemos ver este conjunto de iconos para el *Menú principal*, mostrado en la parte superior izquierda al *Activar edición*:

- *Asignar roles* . Permite asignar los diferentes roles o tipos de usuario para cada uno de los bloques.
- *Mostrar/Ocultar* . Permite indicar si un bloque debe permanecer oculto o por el contrario debe estar visible en la página.
- *Eliminar* . Para eliminar el bloque de la página.
- *Mover hacia abajo, hacia arriba, derecha e izquierda* (,,…). Permiten cambiar la posición del bloque en la página.

Además, para cada bloque pueden aparecer ciertos elementos administrativos específicos mediante los cuales podemos editar su contenido. Por ejemplo, en la imagen anterior, podemos ver como para el *Menú principal* es posible **añadir un recurso** (una etiqueta de texto, una página Web o de texto, un enlace a un fichero o sitio Web…) o una **actividad** (chat, consulta, cuestionarios, foro, encuesta…) mediante los menús desplegables respectivos.

Volviendo a la **página principal del aula virtual**, podemos ver finalmente que en la parte izquierda de la pantalla aparece el menú *Site Administration* (*Administración del sitio*), el cual contiene las diferentes categorías de actividades administrativas que podemos llevar a cabo con Moodle para el aula virtual de forma general (ver figura 5-44).

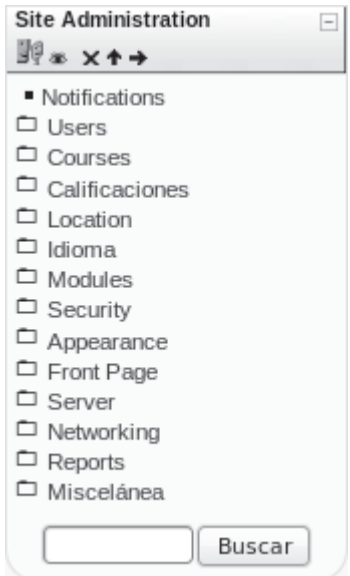


Figura 5-44. Moodle – Administración del sitio

Veamos qué características es posible configurar con cada uno de estos menús en nuestro sitio *e-learning*:

- *Notifications (Notificaciones).* Permite acceder a las notificaciones generadas en nuestro sitio Moodle.

- *Users (Usuarios).* Se encuentra dividida a su vez en tres categorías: *Authentication (Autenticación), Accounts (Cuentas) y Permissions* (*Permisos*). Podemos por tanto administrar el modo en el que se autentican los usuarios en el sitio, gestionar la lista de usuarios, definir roles del sistema, y establecer políticas para los usuarios. Por ejemplo, para la creación de un usuario vamos a la categoría *Accounts* y seleccionamos el enlace *Nuevos usuarios*. De esta forma aparece el formulario para dar de alta un nuevo usuario, que podemos ver en la figura 5-45.

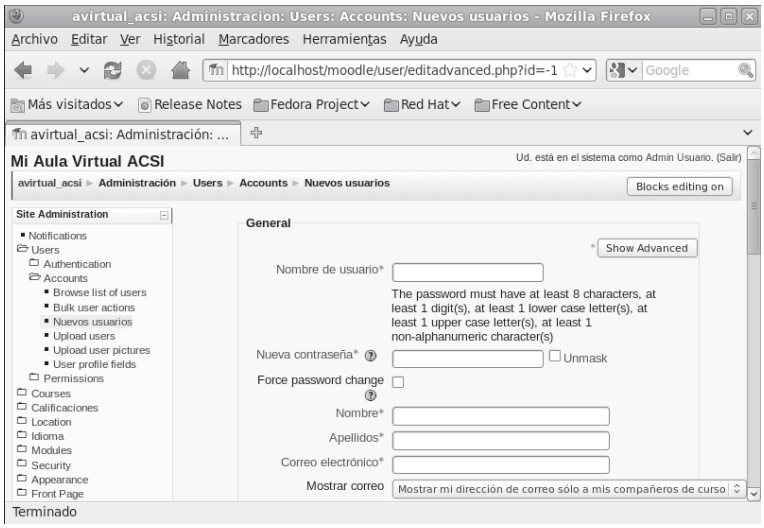


Figura 5-45. Moodle – Nuevos usuarios

- *Courses (Cursos).* Permite añadir cursos o editar los existentes, realizar copias de seguridad o establecer la configuración general por defecto para todos los cursos: formato, número de semanas/temas, número de elementos, etc.

- *Calificaciones.* Permite configurar cómo pueden ser creadas las calificaciones para los cursos: tipos de categorías, elementos que las componen, formato del informe de calificaciones, y otros.

- *Location (Ubicación).* Contiene las opciones para seleccionar la zona horaria y país para el aula virtual.

- *Idioma.* Idioma por defecto, detección automática para el idioma, idiomas instalados, conjuntos de lenguajes a mostrar en los menús, etc.

- *Modules (Módulos).* Contiene la configuración de todos los módulos del aula virtual clasificados en *Actividades* (administración de actividades, chat, cuestionarios, foros, recursos, tareas glosarios…), *Blocks* (*Bloques:* cursos, usuarios *on-line*, *feeds RSS,* secciones con enlaces) y *Filtros* (usados para mostrar u ocultar determinada información en las páginas).

- *Security (Seguridad).* Permite crear políticas de seguridad para el sitio, seguridad HTTP o en los módulos, notificaciones, antivirus,…Por ejemplo, resulta importante determinar qué políticas de usuario vamos a establecer para proteger su privacidad, si queremos publicar nuestra aula virtual en Google, requerir conexiones seguras HTTP en los registros en la página, etc.

- *Appearance (Apariencia).* Permite seleccionar y configurar el tema para el aula virtual, la visualización del calendario, la configuración del editor HTML para los

contenidos… en definitiva todos los aspectos relacionados con el aspecto visual del sitio y las herramientas que podemos utilizar en él.

- *Front page (Página principal).* Configuración para la página principal del aula virtual: nombre y nombre corto para el sitio, la descripción del mismo, número de elementos a mostrar, bloques visibles…

- *Server (Servidor).* Estadísticas, información y configuración sobre el servidor y las aplicaciones que soportan Moodle. Contiene interesantes actividades para medir y optimizar el rendimiento del servidor, los parámetros de ejecución para la aplicación… y opciones de depuración y mantenimiento, manejo de sesiones, etc.

- *Networking (Red).* Clave pública del servidor, si habilitamos o no las operaciones de *Networking* en el servidor, y otras.

- *Reports (Informes).* Copias de seguridad de la información, configuración de los *logs* a realizar en el sistema y para los cursos, es decir, tenemos acceso a un completo conjunto de opciones a configurar para los informes sobre prácticamente cualquier actividad del aula virtual.

- *Miscelánea.* Otras opciones no clasificadas en las categorías anteriores.

5.3.3.2 ADMINISTRACIÓN DE CURSOS

En la parte central de la pantalla tenemos disponible la lista *Cursos disponibles* para el aula virtual. Esta lista además contiene el botón *Añadir un nuevo curso* que nos permite crear y añadir uno nuevo a la lista. Si hacemos clic en este botón accedemos a la página que podemos ver en la figura 5-46.

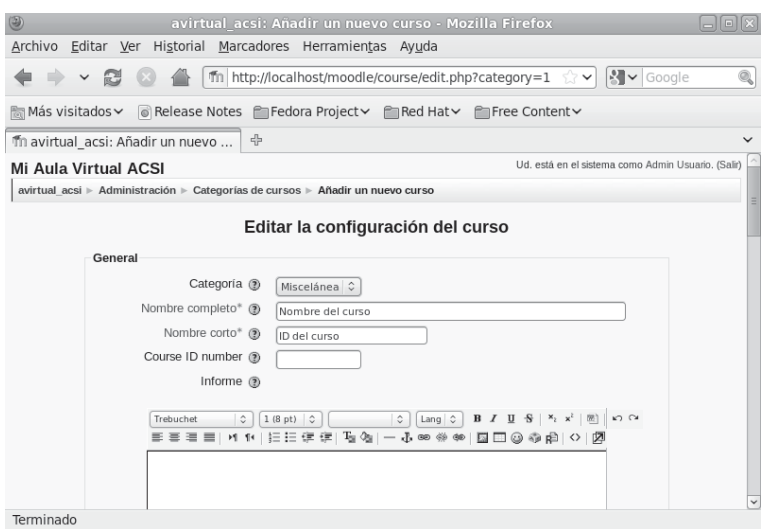


Figura 5-46. Moodle – Creación de un nuevo curso

Debemos escribir entonces la categoría a la que pertenece el curso, el nombre completo y nombre corto para el mismo, un identificador numérico, una breve descripción, el formato para el curso (por temas, semanal…), el número de bloques temáticos o semanas, la fecha de inicio, si mostrar o no calificaciones… es decir, debemos caracterizar por completo el nuevo curso. Otros aspectos que podemos configurar son por ejemplo la accesibilidad al curso o si queremos renombrar los roles para el curso. Pulsamos *Guardar cambios* para crear el curso. Al crearlo debemos después añadir los diferentes participantes al curso para comenzar (esto también lo podemos hacer posteriormente); *asignar los roles.*

Desde este momento es posible entrar a la **página principal del curso** (véase la figura 5-47) y comenzar a editar el curso: bloques de contenido, estructura, actividades, etc.

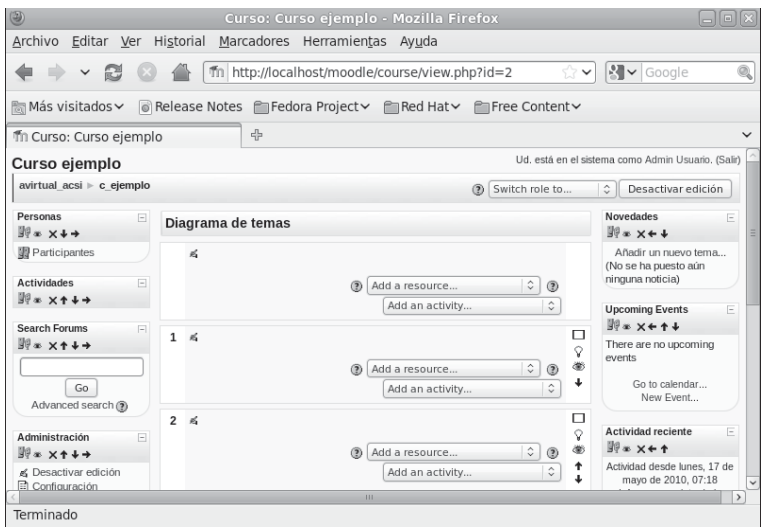


Figura 5-47. Moodle – Página principal de un curso

Entre los bloques creados por defecto se encuentran: *Personas, Actividades, Novedades, Upcoming Events (Próximos eventos)*, *Actividad reciente, Search Forums (Búsqueda en los foros)*, *Administración y Cursos*. Existe además un bloque especial denominado *Blocks*, que contiene un botón *Añadir* que nos permite añadir más bloques a la página, como los que podemos ver en la figura 5-48.

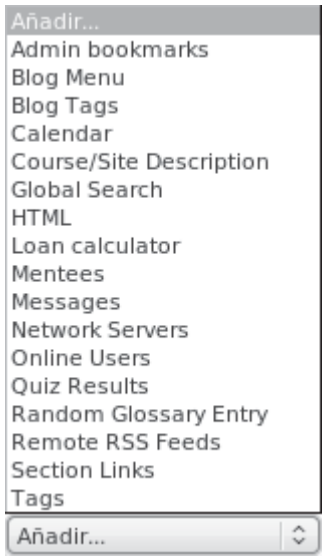


Figura 5-48. Moodle – Añadir un nuevo bloque

El bloque *Administración* es especialmente importante ya que contiene todas las operaciones básicas que podemos realizar para gestionar y administrar el curso. En la figura 5-49 podemos ver las diferentes categorías/enlaces que contiene.

Figura 5-49. Moodle – Bloque Administración para un curso

Vemos de forma breve en qué consiste cada uno de ellos:

- *Desactivar edición.* Oculta todos los elementos disponibles para la edición de la página para los bloques que la componen así como los diferentes bloques que deben permanecer ocultos.

- *Configuración.* Permite acceder a la página de configuración global para el curso, que vimos en la creación del mismo.

- *Assign roles (Asignar roles).* Accediendo a esta página podemos asignar los diferentes usuarios que forman parte del curso agrupados en roles.

- *Calificaciones.* Es usado para tener una vista general de las calificaciones para el alumnado en el curso.

- *Groups (Grupos).* Permite crear y visualizar los grupos de usuarios creados para el curso, así como los diferentes alumnos que los componen.

- *Copia de seguridad.* Permite realizar copias de seguridad del curso.

- *Restaurar.* Es usado para restaurar una copia de seguridad previamente creada para el curso.

- *Import (Importar).* Permite importar actividades y grupos de usuarios desde otro curso al curso actual.

- *Reset (Resetear).* Permite eliminar toda la información sobre los usuarios (usuarios asignados, mensajes en los foros, etc.) en el curso manteniendo las actividades y contenidos del mismo. Podemos elegir qué elementos deben permanecer y cuáles no. Es muy útil para *limpiar* un curso una vez que éste ha finalizado y debe estar accesible para nuevos usuarios.

- *Reports (Informes).* Podemos acceder a informes en tiempo real sobre participación, actividades, por usuario, por actividad, por curso, por fecha…

- *Preguntas.* Permite crear preguntas para un determinado curso: preguntas de opción múltiple, parejas, respuesta corta, numéricas, etc.

- *Archivos.* Podemos crear una estructura de directorios para almacenar los diferentes archivos para un curso.

- *Quitar de.* Con esta opción podemos dejar de pertenecer al curso.

- *Profile (Perfil).* Tal y como puede ver en la figura 5-50 puede ver y modificar la información general del usuario. Además, si somos administradores del sitio y vemos la ficha de un usuario podremos actualizar su información, utilizar su cuenta de usuario, etc.

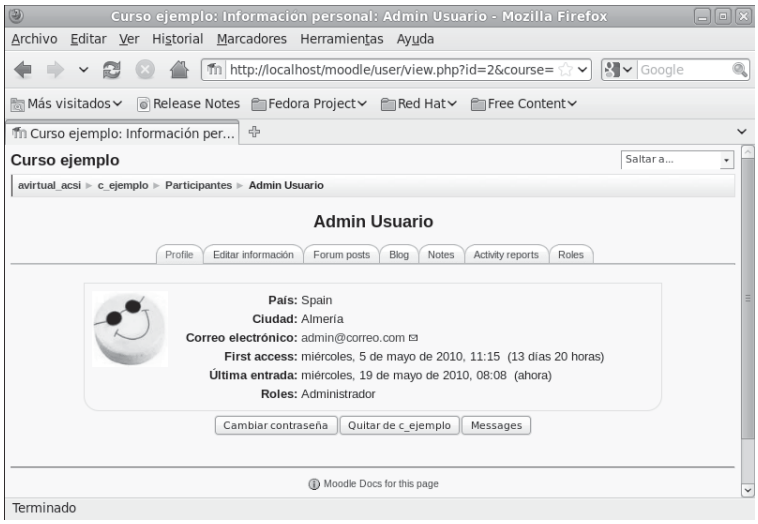


Figura 5-50. Moodle – Perfil de usuario

Un elemento muy importante de Moodle es el menú desplegable que nos permite **cambiar entre los diferentes roles**. De esta forma, cuando creemos un curso podemos cambiar de rol para ver la apariencia y funcionalidad del curso. Por ejemplo puede cambiar al rol de *Estudiante* para comprobar el correcto funcionamiento del curso.

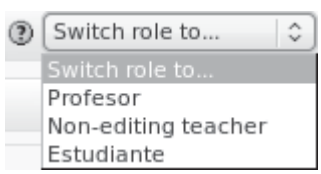


Figura 5-51. Moodle – Cambiar rol

Como podemos ver en la gran variedad de características y opciones a configurar Moodle es una plataforma *e-learning* completamente configurable y adaptable a nuestras necesidades y preferencias. Esto, unido a la gran eficiencia en su ejecución y funcionalidad ha provocado que tenga una gran difusión y aceptación por la comunidad educativa.

5.4 MEDIAWIKI

De sobra es conocido por todos el portal Wikipedia (http://es.wikipedia.org). MediaWiki (http://www.mediawiki.org/wiki/MediaWiki/es) es el software que originalmente fue escrito para Wikipedia, y que ahora es utilizado como plataforma para el lanzamiento de prácticamente cualquier wiki. Una wiki es un entorno colaborativo en la que los usuarios pueden redactar artículos e incluir información tratando temas diversos.

5.4.1 Requisitos previos

Antes de iniciar el proceso de instalación debemos crear la base de datos para el sitio y establecer los privilegios a un determinado usuario.

```
# mysql –u root -p
mysql> create database wikidb;
mysql> quit;
```

Para que MediaWiki pueda acceder al contenido de la base de datos creamos el usuario wikiuser con todos los privilegios sobre la base de datos:

```
mysql> GRANT ALL PRIVILEGES ON wikidb.* TO wikuser@localhost
IDENTIFIED BY 'contrasena';
mysql> flush privileges;
mysql> quit;
```

5.4.2 Instalación

Cumplidos los requisitos previos para después poder realizar su configuración correctamente, instalamos MediaWiki desde repositorios mediante *yum install* (versión 1.15.1):

```
# yum install mediawiki
```

Para la instalación es necesario descargar aproximadamente 73 Mb, actualizar paquetes y dependencias, instalar nuevos módulos para PHP, etc., por lo que el proceso suele durar algunos minutos.

Además, es necesario instalar manualmente el paquete *mailcap* por lo que ejecutamos

```
# yum install mailcap
```

Como para el resto de aplicaciones de nuestro servidor LAMP es necesario para poder operar con MediaWiki que tanto el servicio Apache (*httpd*) como MySQL (*mysqld*) se encuentren iniciados (si ya fue configurado que se iniciaran automáticamente al arrancar el sistema, los reiniciamos con *restart* para que tomen los cambios realizados):

```
# service mysqld start
# service httpd start
```

MediaWiki se instala en el directorio */usr/share/mediawiki*. Este directorio no es accesible por Apache y lo recomendable es copiarlo al directorio */var/www/html* ejecutando:

```
# cp /var/www/wiki /var/www/html/wiki –R
```

También disponemos de los temas de interfaz (diseño y ficheros de estilo CSS, ficheros javascript, e imágenes para la wiki) en el directorio */usr/share/mediawiki/skins*. Los copiamos en el directorio */var/www/html/wiki*:

```
# cp /usr/share/mediawiki/skins /var/www/html/wiki -R
```

A continuación escribimos en el navegador http://localhost/wiki/ para acceder a nuestra wiki (véase la figura 5-52) y realizamos los siguientes pasos para completar el proceso de instalación:

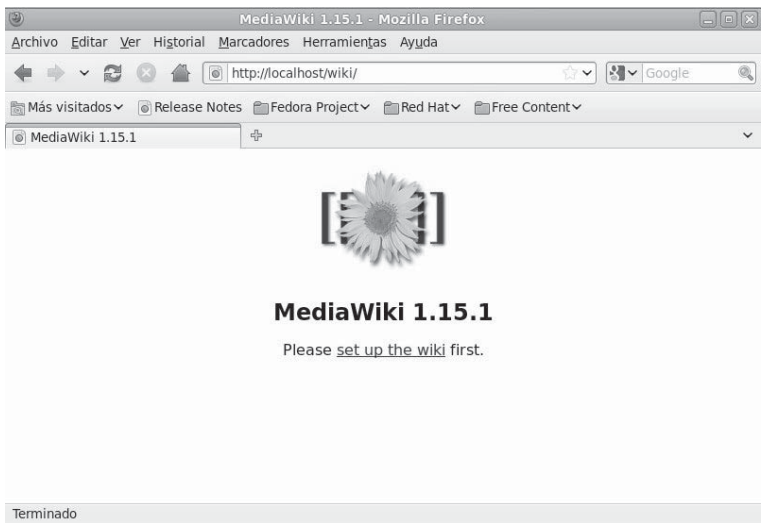


Figura 5-52. MediaWiki – Comienzo de la instalación

- Pulsamos en *set up the wiki* para iniciar el proceso de instalación.

- Como podemos ver en la figura 5-53, el servidor web no dispone de permiso de escritura para el subdirectorio de configuración. Para solucionarlo ejecutamos los siguientes comandos para poder otorgarle estos permisos y proseguir con la instalación:

```
# cd /var/www/html/wiki
# chmod a+w config
```

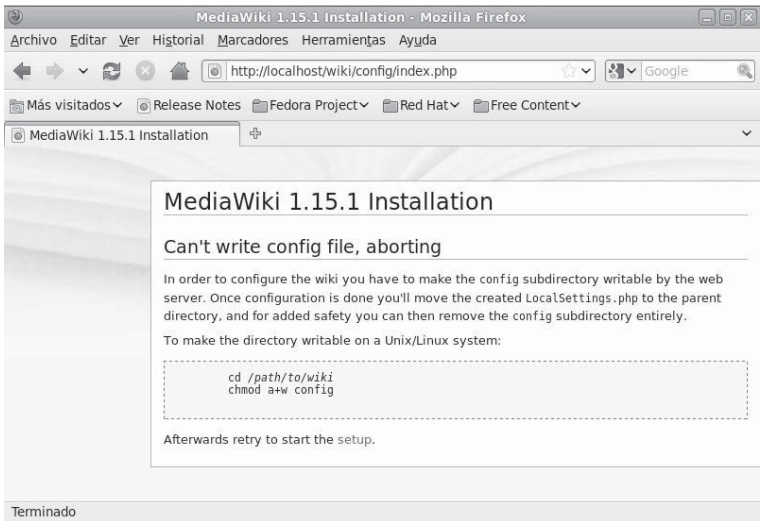


Figura 5-53. MediaWiki – Otorgando permisos de escritura al fichero config

- Hacemos clic en *setup* para continuar con la instalación (véase la figura 5-54).

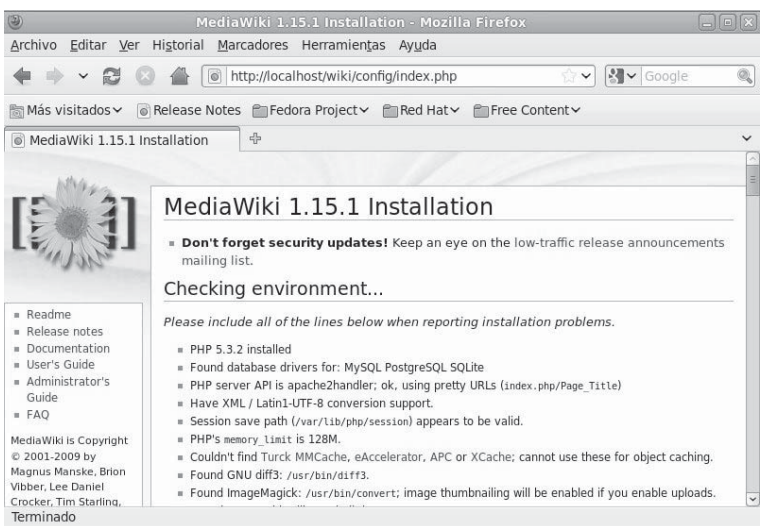


Figura 5-54. MediaWiki – Instalación – Comprobando requisitos

- El sistema comprueba los requisitos para realizar la instalación y si son correctos nos solicita la información sobre la wiki (nombre de la wiki, email de contacto, idioma, etc.).

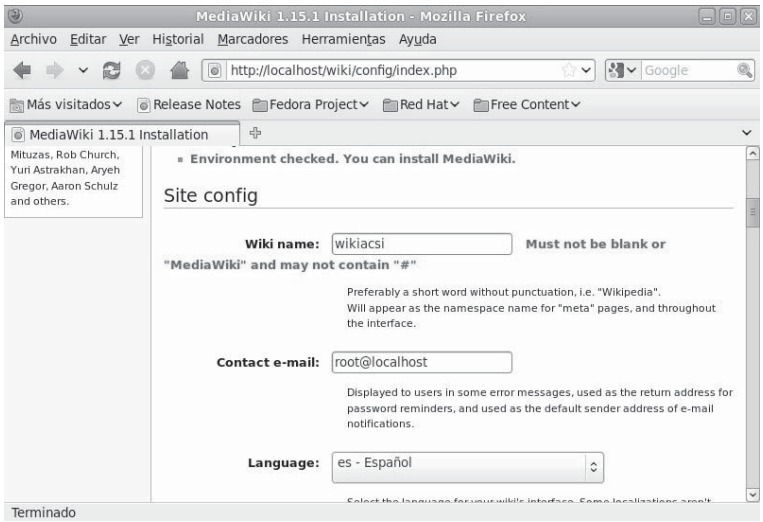


Figura 5-55. MediaWiki – Instalación – Configuración del sitio

- Seleccionamos el tipo de licencia para el contenido de la wiki e indicamos el login del administrador y su contraseña.

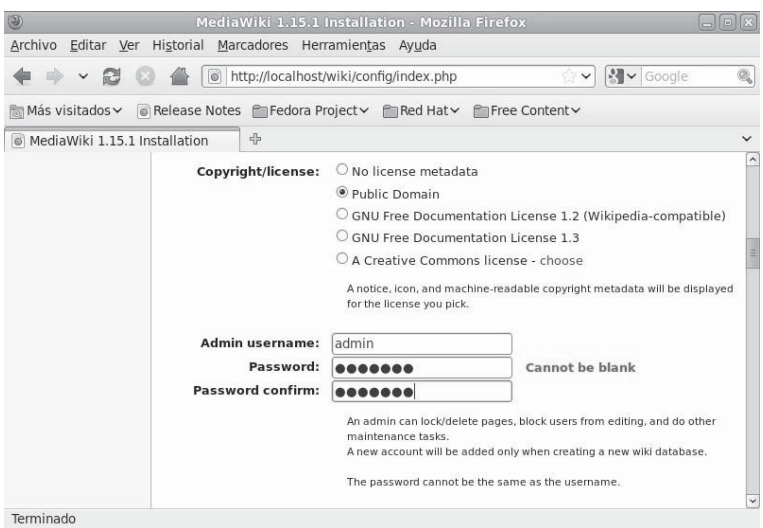


Figura 5-56. MediaWiki – Instalación – Configuración de la cuenta del administrador

- Configurar la conexión de la wiki a la base de datos que hemos creado anteriormente. Indicamos el tipo de base de datos (MySQL), el equipo en el que se encuentra (localhost), el nombre de la base de datos y los datos de la cuenta de usuario para poder utilizarla.

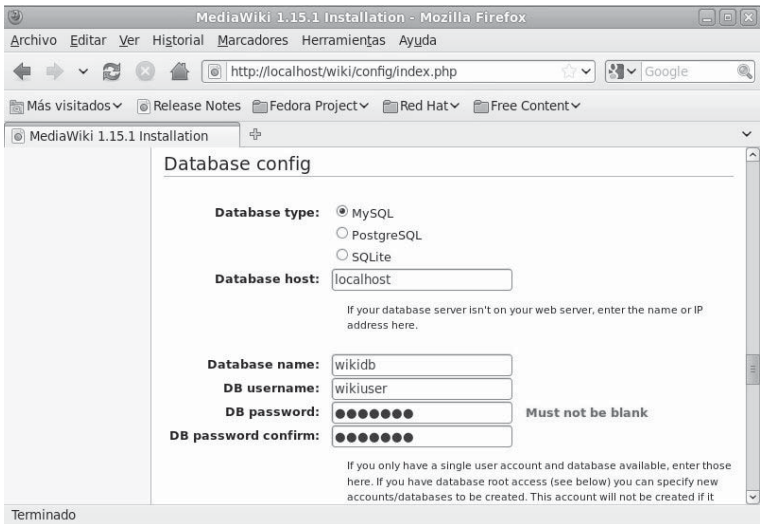


Figura 5-57. MediaWiki – Instalación – Configuración de la base de datos

- Para finalizar pulsamos el botón *Install MediaWiki*.

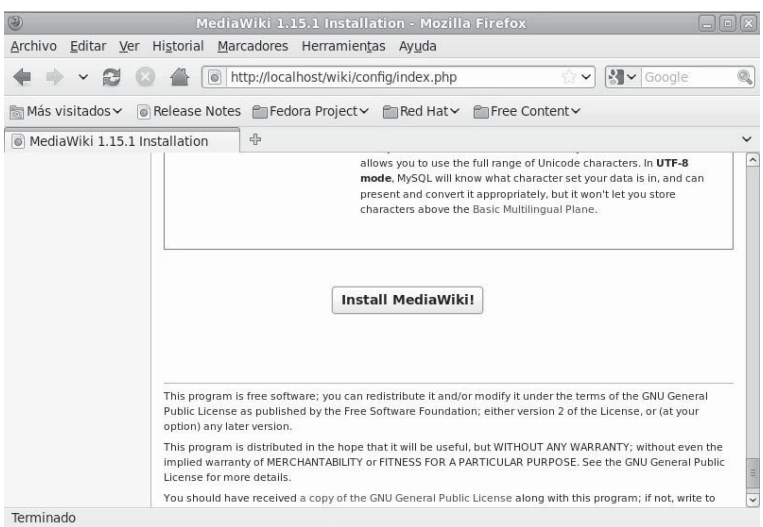


Figura 5-58. MediaWiki – Instalación – Ejecución

- Una vez finalizada la instalación el sistema nos muestra un mensaje de confirmación.

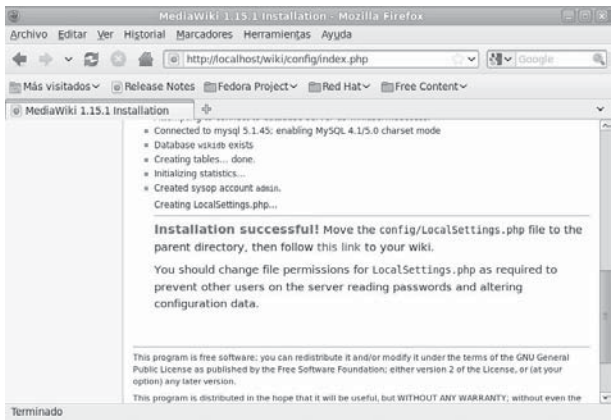


Figura 5-59. MediaWiki – Instalación – Fin

Para finalizar debemos copiar el fichero *LocalSettings.php* localizado en el directorio */var/www/html/wiki/config* al directorio */var/www/html/wiki*:

```
# cp /var/www/html/wiki/config/LocalSettings.php /var/www/html/wiki
```

Eliminamos, por motivos de seguridad, el subdirectorio config original:

```
# rm –fr /var/www/html/wiki/config
```

Si pulsamos en el link proporcionado en el último paso de la instalación o escribimos en el navegador http://localhost/wiki podemos comprobar que ya tenemos el acceso configurado a nuestra wiki y podemos comenzar a trabajar con ella:

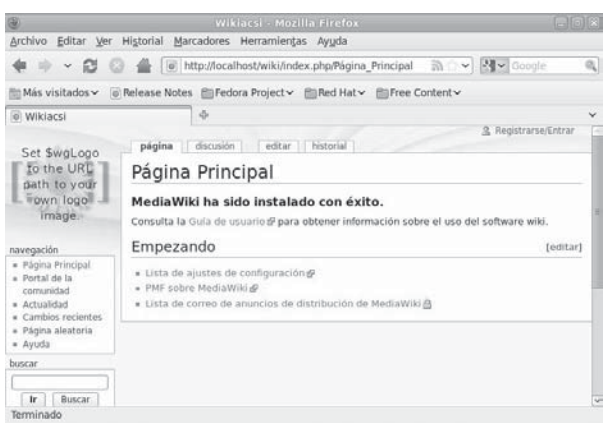


Figura 5-60. MediaWiki – Página principal de la wiki

Desde ahora podemos comenzar a configurar nuestra wiki para adaptarla completamente a las necesidades que tengamos. Podemos comenzar a dar de alta usuarios, incluir una imagen para la misma, crear la página principal, etc.

Más información

Página web oficial de MediaWiki
http://www.mediawiki.org

5.4.3 Personalización

Para comenzar a personalizar nuestra *wiki* abrimos el navegador y escribimos la dirección http://nombre_del_sitio/wiki (o también podemos escribir http://localhost/wiki si nos encontramos en el mismo equipo en el que está instalado MediaWiki). Así aparece la página principal de la *wiki* que podemos ver en la figura 5-61.

Para comenzar a personalizar el aspecto y contenido de la *wiki* o para administrarla debemos **entrar en el sitio** como usuario administrador, por lo que hacemos clic en el enlace *Registrarse/Entrar* situado en la esquina superior derecha de la pantalla e introducimos las credenciales del usuario *admin* en el formulario de acceso.

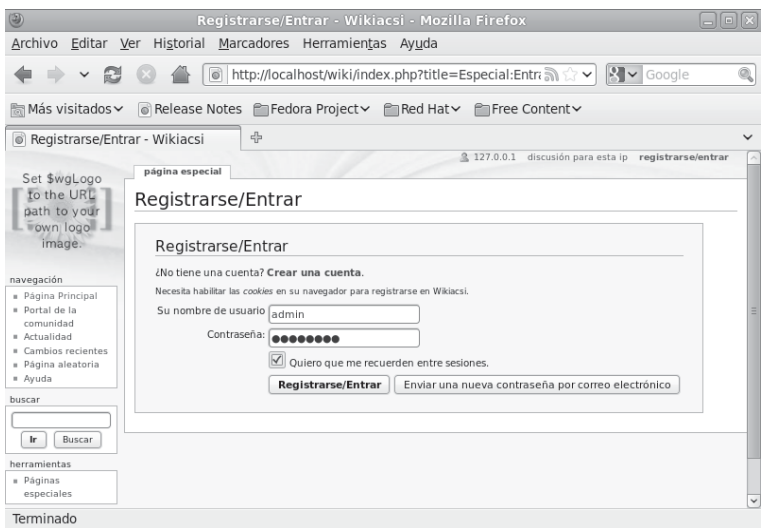


Figura 5-61. MediaWiki – Registrarse/Entrar

Pulsamos el botón *Registrarse/Entrar*. Veamos a continuación los dos primeros pasos necesarios para comenzar a personalizar el aspecto, el comportamiento y funcionalidad de la MediaWiki.

5.4.3.1 CAMBIAR EL LOGOTIPO

Para cambiar el logotipo de la wiki debemos realizar dos pasos:

1. En primer lugar debemos elegir la imagen que podemos mostrar como logotipo y ubicarla en el servidor. Por defecto el archivo que contiene la imagen es ubicado en el directorio */wiki/skins/common/images*, aunque podemos hacerlo en cualquier otro que queramos. El tamaño para la imagen debe ser el adecuado para la zona en la que se va a mostrar.
2. Una vez que disponemos de la imagen en el servidor, editamos la variable *$wgLogo* en el fichero */var/www/html/wiki/LocalSettings.php* y escribimos en ella la ruta a la imagen, por ejemplo:

```
$wgLogo =
"{$wgScriptPath}/skings/common/images/logoMIRADA.jpg";
```

donde la variable $wgScriptPath recoge la ruta al directorio /wiki.

Si guardamos los cambios en el fichero y actualizamos la página principal de la *wiki* podemos ver como ahora carga la nueva imagen para el logotipo (véase la figura 5-62).

Figura 5-62. MediaWiki – Cambiar el logotipo

5.4.3.2 PERMITIR LA SUBIDA DE ARCHIVOS

Es habitual que sea necesario que los usuarios puedan subir ficheros al servidor para enlazarlos en sus artículos publicados en la *wiki*: imágenes, documentos, audio, etc. Esta opción se encuentra deshabilitada por defecto al instalar MediaWiki por motivos de seguridad, por lo que para habilitarla debemos modificar la variable *$wgEnableUploads* del fichero de configuración */var/www/html/wiki/LocalSettings.php* de la siguiente forma:

```
$wgEnableUploads = true;
```

En MediaWiki no es posible subir archivos de forma anónima, por lo que para poder hacerlo debemos estar registrados en el sitio. Por defecto los tipos de fichero que es posible subir son los siguientes: *png, gif, jpg y jpeg*. Una vez que tenemos configurada la subida de archivos podemos acceder a la misma a través del enlace *Subir archivo* en el menú *Herramientas*, situado en la columna izquierda de la pantalla. Así, aparece la ventana *Subir archivo* (véase la figura 5-63), en la que *examinamos* en busca del fichero a subir y

escribimos un nombre y descripción para el mismo. Finalmente pulsamos el botón *Subir un archivo*.

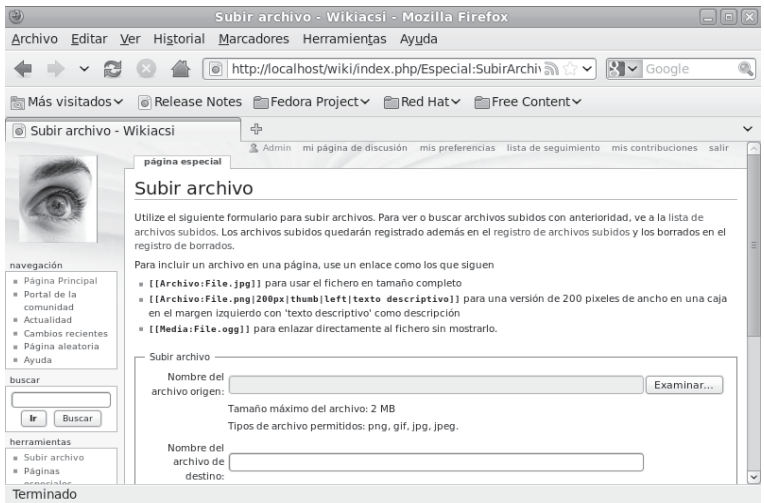


Figura 5-63. MediaWiki – Subir archivo

5.4.3.3 ESTABLECER LOS TIPOS DE FICHEROS DE SUBIDA

Si queremos permitir la subida de ficheros de un tipo diferente a cualquiera de los permitidos por defecto, por ejemplo un fichero comprimido *.zip o .rar*, un documento *.pdf*, etc., debemos incluir o editar la variable *$wgFileExtensions* de la siguiente manera:

```
$wgFileExtensions = array( 'png', 'gif', 'jpg', 'jpeg', 'svg',
'mp3', 'ogg', 'avi', 'swf', 'pdf', 'zip', 'doc', 'xls' );
```

Si accedemos a la ventana *Subir archivo* de nuevo, podemos ver en la información mostrada como ahora los tipos de archivos permitidos son los que hemos incluido en la nueva configuración:

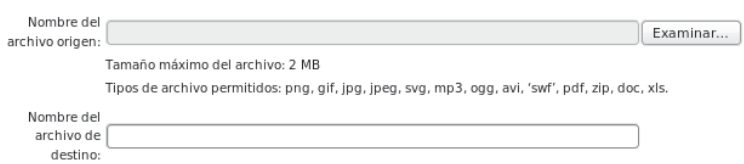


Figura 5-64. MediaWiki – Establecer los tipos de ficheros de subida

5.4.3.4 CONFIGURAR LA ZONA HORARIA

Resulta importante ajustar la zona horaria del servidor para que se muestre correctamente la hora de creación/modificación de los contenidos.

Para configurar la zona horaria deberemos editar el fichero de configuración *LocalSettings.php* y poner al final el siguiente código:

```
# Ajustar zona horaria
$wgLocaltimezone = "Europe/Madrid";
$oldtz = getenv("TZ");
putenv("TZ=$wgLocaltimezone");
$wgLocalTZoffset = date("Z") / 60;
putenv("TZ=$oldtz");
```

Como podemos ver se ha especificado la zona Europe/Madrid. Si se desea establecer una zona horaria diferente se pueden ver los códigos de las diferentes zonas en la dirección http://en.wikipedia.org/wiki/List_of_zoneinfo_time_zones.

5.4.4 Utilización

A continuación vamos a ver cómo crear una nueva página en MediaWiki así como los aspectos más generales de la administración de las mismas. Para añadir una nueva página, una vez que hemos sido autenticados con nombre de usuario y contraseña, normalmente lo hacemos a través de un enlace que ha sido creado en otra de las páginas de la wiki, editándola. Para editar/crear una página que estamos viendo hacemos clic en la pestaña superior *Editar* o *Crear*, según proceda. Por ejemplo, hacemos clic en *Editar* para nuestra página principal, por lo que aparece el editor para el contenido de la misma, como podemos ver en la figura 5-65.

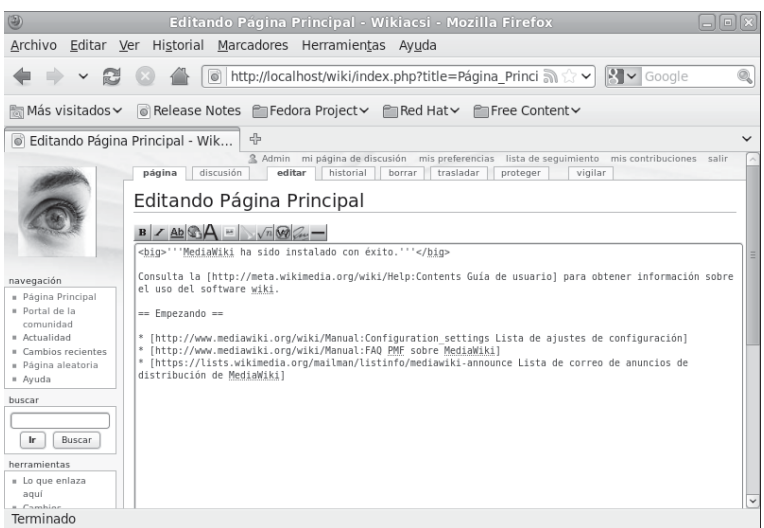


Figura 5-65. MediaWiki – Editar

En la parte central de la ventana aparece un cuadro de texto en el cual podemos añadir el contenido de la página. Para ello además disponemos de una pequeña barra de iconos que nos permiten llevar a cabo las operaciones más básicas para la creación del contenido:

Figura 5-66. MediaWiki – Barra de iconos para la edición

Mediante su uso podemos cambiar el tipo/tamaño de la fuente del texto que hemos escrito, establecer un título para un apartado, insertar una imagen…; quizás una de las opciones más importantes sea la de insertar un enlace, externo o interno. Los enlaces externos contienen URL para sitios Web externos a la wiki, mientras que los internos hacen referencia a otras páginas de la misma. Al crear un enlace interno escribimos el nombre de la página a la que queremos hacer referencia (véase la figura 5-67), o bien el nombre de una nueva página si no existe, por lo que posteriormente haciendo clic en ese enlace podemos crearla.

Figura 5-67. MediaWiki – Creación de un enlace interno/nueva página

Además, en la parte inferior de la ventana de edición de la web disponemos de un conjunto de opciones para la misma, como para escribir un resumen breve de la misma, marcarla como una edición menor, *vigilar* la página… y los botones para *Grabar la página, Mostrar previsualización* del contenido creado o *Mostrar cambios* (respecto a otras versiones de la misma). Al guardar la página, podemos ver como los enlaces a las páginas que no existen aún se encuentran en rojo (haciendo clic en ellos podemos comenzar a crearlas), como podemos ver en la figura 5-68.

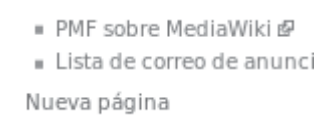


Figura 5-68. MediaWiki – Enlace interno a una página por crear

Para administrar las páginas que ofrece así como la funcionalidad mostrada por MediaWiki lo hacemos a través del enlace *Páginas especiales* situado en el cuadro *Herramientas* de la izquierda en la ventana principal. Cuando hacemos clic en él aparece la pantalla que vemos en la figura 5-69.

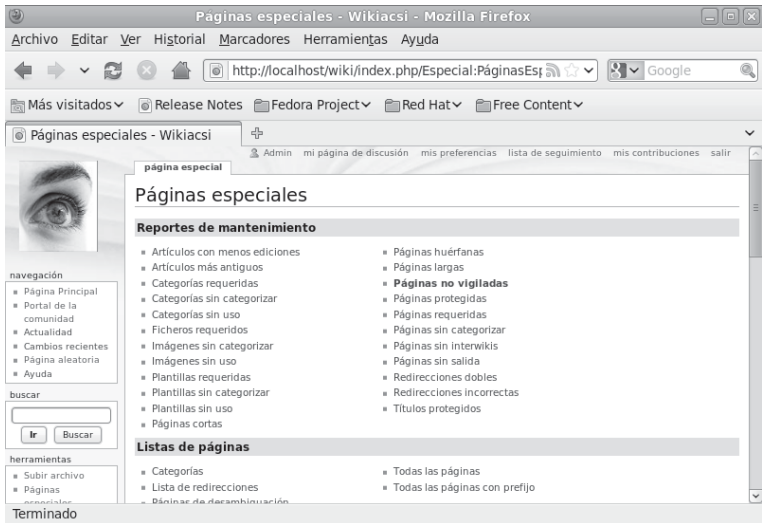


Figura 5-69. MediaWiki – Páginas especiales

Desde aquí disponemos de numerosos enlaces a las operaciones administrativas que tenemos disponibles así como a determinadas páginas de importancia para la wiki, agrupados por categorías:

- *Reportes de mantenimiento*. Nos permite acceder a diferentes informes en relación a los objetos de nuestra wiki: artículos, imágenes, categorías, ficheros, plantillas, páginas, redirecciones, etc. Por ejemplo podemos ver cuáles son los artículos con menos ediciones o los más antiguos, las imágenes que están sin categorizar o en desuso, o las páginas huérfanas, protegidas… Todos los informes son bastante útiles ya que ofrecen un medio importante para el mantenimiento del contenido de la wiki.
- *Listas de páginas*. Permite administrar las diferentes categorías disponibles en la wiki así como obtener listados de páginas y redirecciones.
- *Registrarse/Entrar*. Abre la página que contiene el formulario para registrarnos/entrar en la wiki.
- *Usuarios y permisos*. Permite administrar los usuarios y los permisos, ver sus contribuciones, etc. También es posible establecer permisos para grupos de usuarios y listar las direcciones IP o usuarios bloqueados. Desde aquí podemos también editar nuestras *Preferencias* como usuario (p.e. apariencia, tipo de edición) tal y como podemos ver en la figura 5-70.

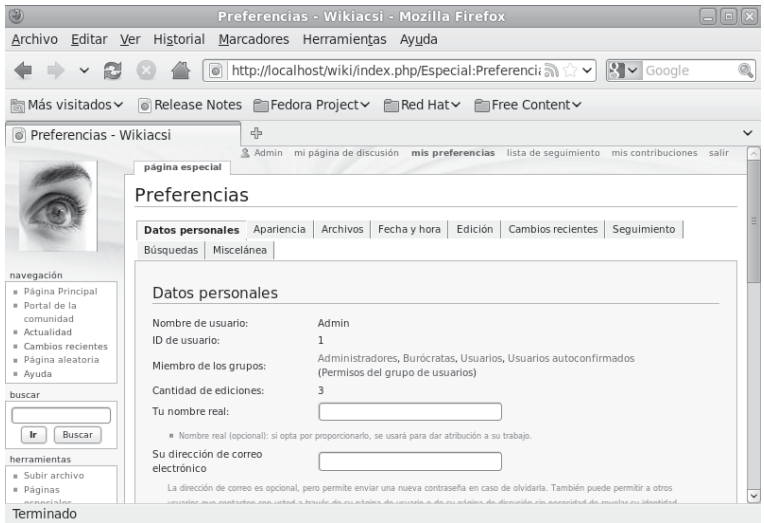


Figura 5-70. MediaWiki – Preferencias de usuario

- *Cambios recientes y registros*. Nos permite monitorizar y seguir los diferentes cambios que han tenido lugar en la wiki. Por ejemplo, podemos ver cuáles han sido las últimas páginas creadas o conocer los cambios registrados.
- *Páginas sobre archivos*. Permite realizar operaciones muy importantes por ejemplo para conocer la ruta completa a un archivo en la wiki o la existencia de ficheros duplicados en el sistema. También enlaza a las páginas para la subida de archivos y listado completo de los mismos.
- *Datos sobre la wiki y herramientas*. Permite acceder a ver un listado con todos los mensajes generados por MediaWiki en estilo *log* (por ejemplo los movimientos de páginas), estadísticas (sobre páginas, ediciones, usuarios, visitas), la versión de la wiki o las páginas más populares.
- *Búsquedas y redirecciones*. Podemos realizar una búsqueda detallada en los diferentes objetos de la wiki, ver un listado de los enlaces externos registrados, visitar una *página aleatoria*, etc.
- *Páginas sobre usos*. Esta categoría también resulta bastante interesante pues proporciona informes con datos e información acerca del uso de artículos, categorías, imágenes, plantillas o páginas (con más ediciones, más enlazados/as, etc.).
- *Herramientas de páginas*. Contiene operaciones para administrar las páginas de la wiki: importar/exportar páginas, restaurar una página borrada anteriormente, o ver las páginas que enlazan a otras páginas (*Lo que enlaza aquí*).
- *Otras páginas especiales*. Permite localizar fuentes de libros a partir de su ISBN.

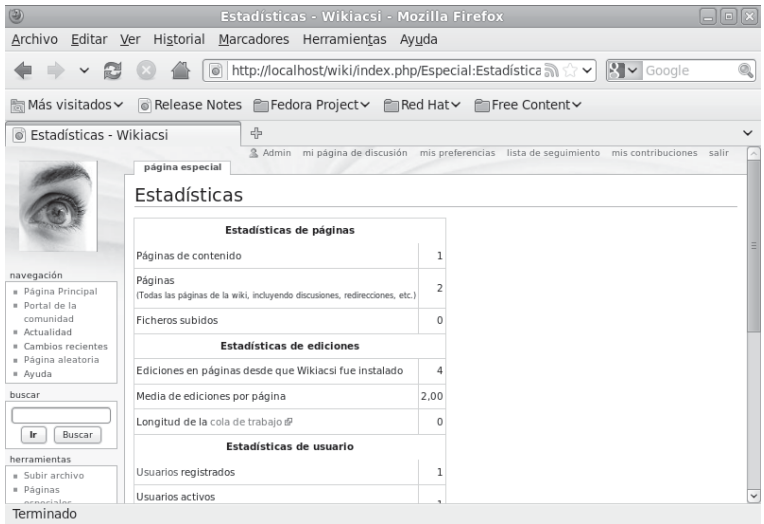


Figura 5-71. MediaWiki – Estadísticas

Como vemos, las posibilidades de personalización y administrativas de MediaWiki son lo suficientemente completas como para crear un sitio totalmente adaptado a nuestro gusto y necesidades.

5.5 WORDPRESS

WordPress (http://wordpress.org/) es una aplicación que actúa como plataforma de publicación *on-line* (por ejemplo, *blogs* o bitácoras) con un cuidado lado visual manteniendo el compromiso por la usabilidad experimentada por el usuario y los estándares Web.

5.5.1 Requisitos previos

Antes de iniciar el proceso de instalación debemos crear la base de datos donde se va a almacenar la información de WordPress.

```
mysql –u root -p
mysql> create database wpdb;
mysql> quit;
```

Establecemos todos los permisos al usuario *wpuser* para que pueda acceder completamente a la base de datos *wpdb*:

```
mysql> GRANT ALL PRIVILEGES ON wpdb.* TO wpuser@localhost IDENTIFIED
BY 'contrasena';
mysql> flush privileges;
mysql> quit;
```

5.5.2 Instalación

Para iniciar el proceso de instalación nos vamos al directorio del sitio web Apache (por defecto */var/www/html*):

```
# cd /var/www/html
```

Y descargamos la última versión de WordPress de la página de descargas: http://wordpress.org/download/.

Descomprimimos el paquete que nos hemos descargado utilizando el comando tar:

```
# tar xvfz wordpress-2.9.2-es_ES.tar.gz
```

Este proceso crea el directorio */var/www/html/wordpress* donde se encuentran todos los ficheros de la aplicación.

A continuación realizamos una copia del fichero de ejemplo de configuración para poder utilizarlo:

```
# cp wp-config-sample.php wp-config.php
```

Y editamos el fichero para indicarle los parámetros de configuración de la base de datos (dirección del servidor, nombre de la base de datos, nombre de usuario y contraseña). Estos valores se crearon en el apartado de requisitos previos.

```
# gedit wp-config.php
```

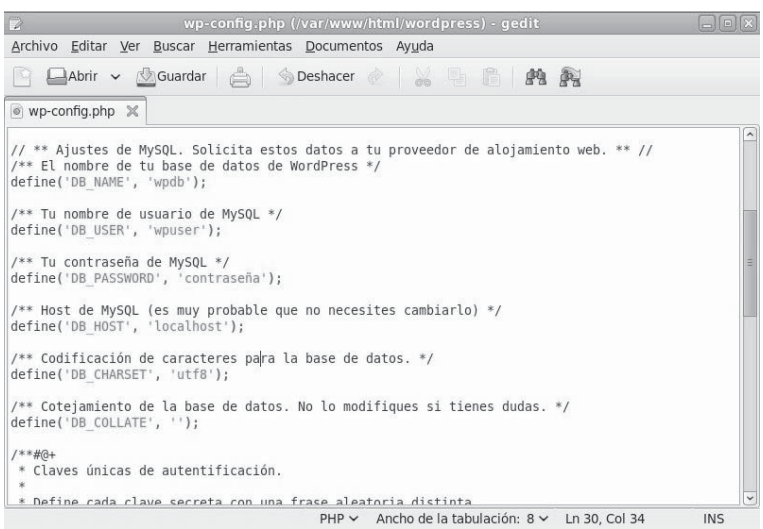


Figura 5-72. WordPress – Fichero wp-config.php – Base de datos

Editamos de esta forma las siguientes líneas:

```
define(DB_NAME, 'wpdb');   //Nombre de la BD
define(DB_USER, 'wpuser'); //Nombre del usuario para la BD
define(DB_PASSWORD, 'contrasena'); //Contraseña del usuario
define(DB_HOST, 'localhost');  //Dirección del servidor de BD
```

Además debemos incluir en este mismo fichero las claves únicas de autenticación para asegurar una encriptación mejor de la información almacenada en las cookies de los usuarios. Para generar las claves ingresamos en el siguiente enlace: https://api.wordpress.org/secret-key/1.1/. Con las claves generadas por WordPress on-line, las copiamos y pegamos en nuestro fichero *wp-config.php*:

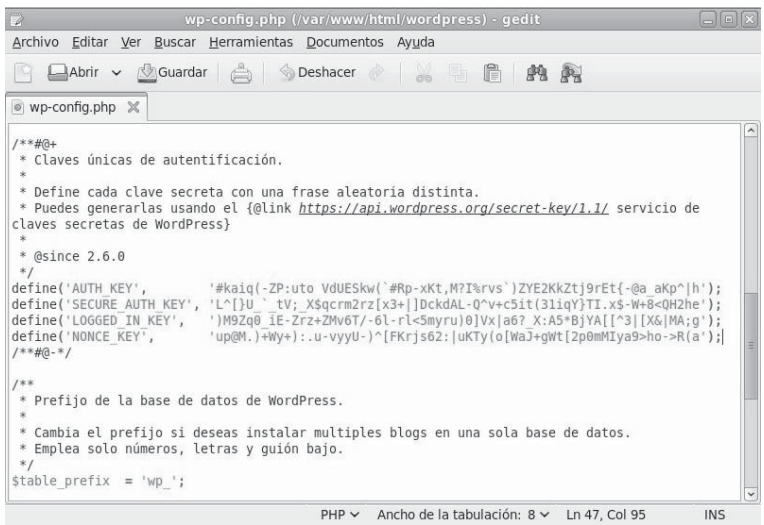


Figura 5-73. WordPress – Fichero wp-config.php – Autenticación

Guardamos los cambios en el fichero y salimos del editor de textos. Ahora sólo queda acceder a través del navegador al fichero *install.php* ubicado en el directorio */ruta/a/wordpress/wp-admin/install.php* y seguir los pasos de instalación. En nuestro caso abrimos el navegador e introducimos en la barra de direcciones http://localhost/wordpress/wp-admin/install.php (véase la figura 5-74).

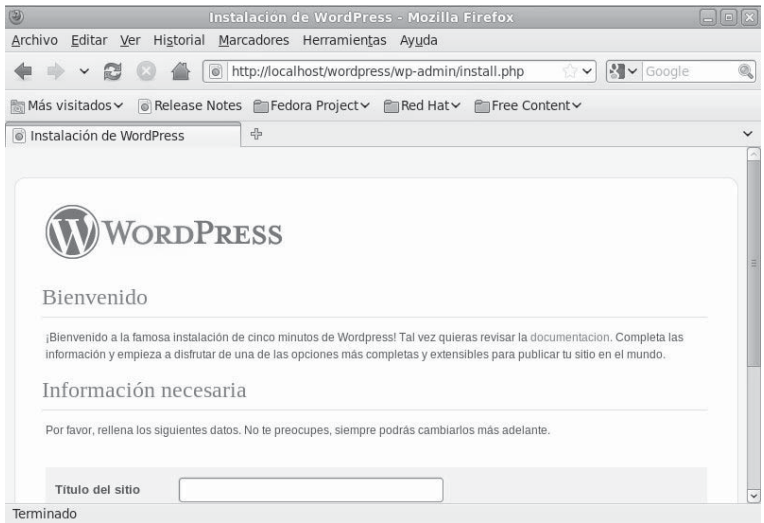


Figura 5-74. WordPress – Instalación – Configuración del sitio (I)

Comenzamos escribiendo el nombre para nuestro sitio y el correo electrónico del administrador. Si queremos podemos habilitar que nuestro sitio aparezca en motores de búsqueda, como Google:

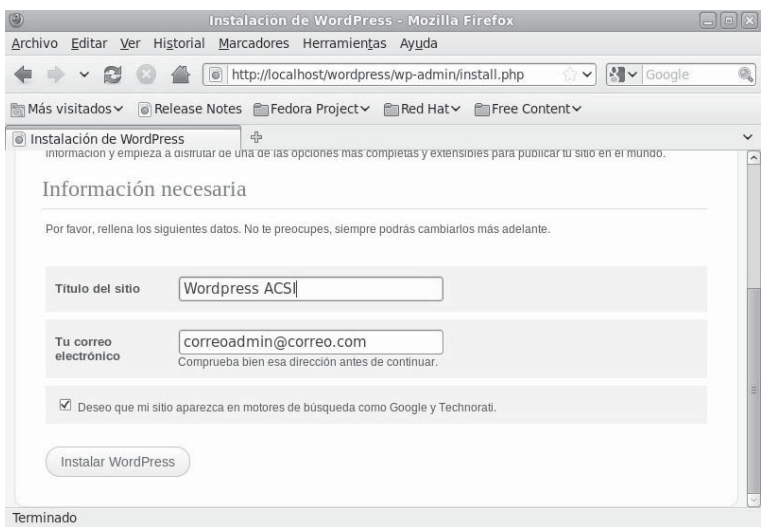


Figura 5-75. WordPress – Instalación – Configuración del sitio (II)

Pulsamos el botón *Instalar WordPress* para ejecutar la misma. Ahora, con la instalación terminada, la web de nuestro sitio WordPress nos ofrece el nombre de usuario y

contraseña correspondientes al administrador del sitio, que debemos anotar. Hacemos clic en el botón *Iniciar sesión* después.

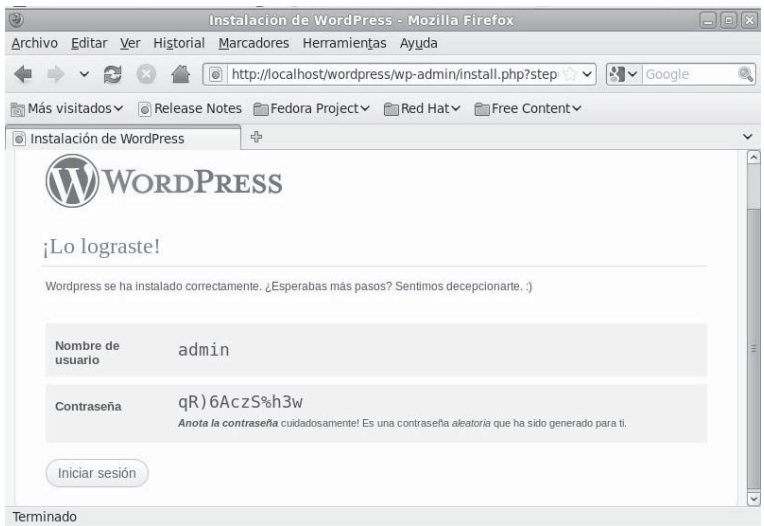


Figura 5-76. WordPress – Información de la cuenta del administrador

Tal y como se muestra en la figura 5-77, iniciamos sesión con los datos de inicio que nos ha facilitado el sistema (véase la figura 5-76).

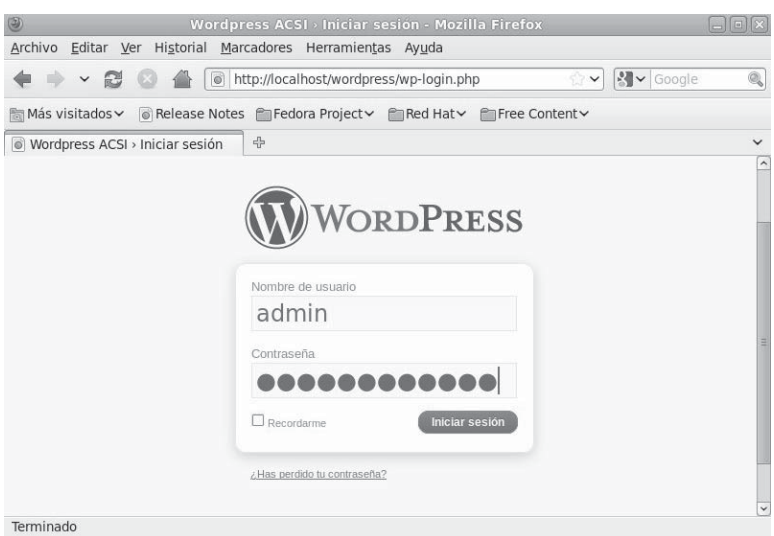


Figura 5-77. WordPress – Inicio de sesión

Ya nos encontramos en nuestra ventana principal de administración de WordPress. Desde aquí podemos gestionar las entradas de nuestro *blog*, administrar las páginas que contiene, realizar una nueva publicación, etc.

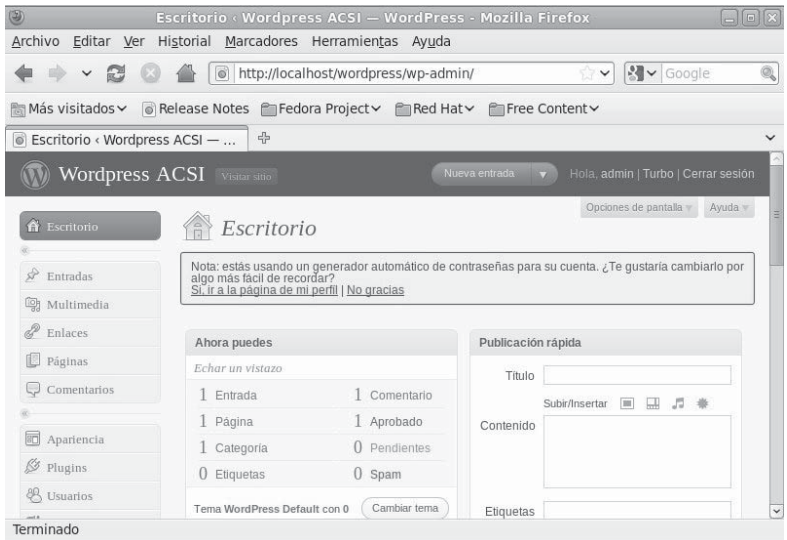


Figura 5-78. WordPress – Escritorio de administración del sitio

Más información

Página web oficial de WordPress
http://codex.wordpress.org/Main_Page.

5.5.3 Utilización

El acceso como visitante a nuestra bitácora creada con WordPress es a través de la dirección http://nombre_de_equipo/wordpress/ (o http://localhost/wordpress/ si nos encontramos en el mismo equipo en el que está instalada). Si nos fijamos en la figura 5-79, la estructura básica de la misma *por defecto* se compone de una cabecera con el título de la misma (fondo azul), un espacio para las entradas publicadas (de la más reciente a la más antigua) y sus comentarios asociados, así como diversos enlaces en la parte derecha para ver la información pública del sitio y filtrar las entradas mostradas: realizando búsquedas, por categorías, etc.

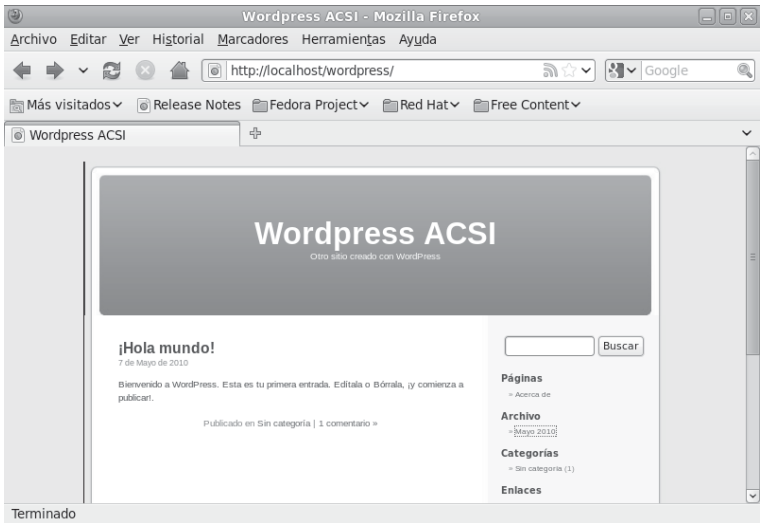


Figura 5-79. WordPress – Escritorio de administración del sitio

Para editar estos aspectos así como configurar otros muchos debemos acceder a la parte administrativa de WordPress a través del enlace http://nombre_del_sitio/wordpress/wp-login.php (o http://localhost/wordpress/wp-login.php). Esta página muestra el formulario de acceso al sitio, en el que introducimos el nombre de usuario (*admin*) y la contraseña que nos ha sido proporcionada en la instalación (véase la figura 5-80).

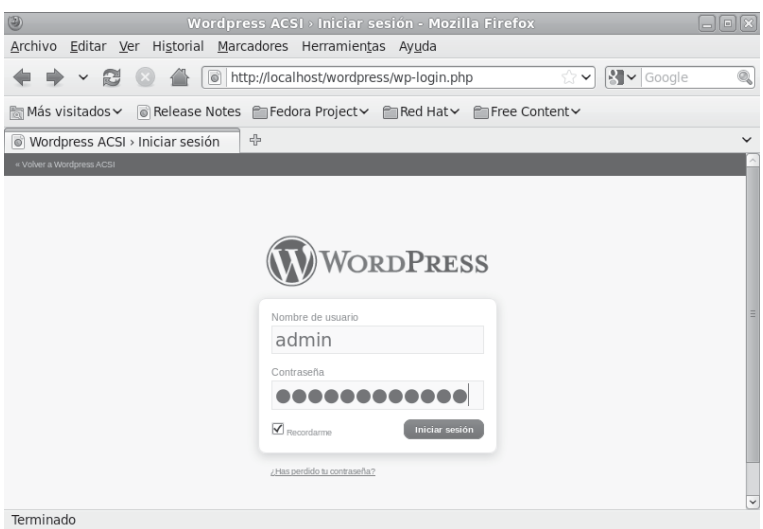


Figura 5-80. WordPress – Acceso a la administración del sitio

Tras pulsar *Iniciar sesión* y haber sido autenticados aparece el *Escritorio* del administrador de WordPress. En la parte central tenemos disponible información general sobre el estado y operaciones recientes en el sitio. Podemos visualizar los últimos comentarios registrados en nuestras entradas, consultar los borradores que dejamos pendientes de publicar, enlaces a noticias de WordPress y su blog oficial y acceder a descarga directa de los *plugins* más utilizados por los usuarios. Sin embargo, las dos opciones más usadas en esta parte central por los administradores son las referentes a la publicación rápida de entradas y la visualización de información sobre entradas, páginas, categorías y etiquetas, que podemos ver en la figura 5-81.

Figura 5-81. WordPress – Publicación rápida y Ahora puedes

Para realizar una publicación rápida simplemente escribimos el título para la misma, editamos el contenido (insertando texto o cualquier objeto multimedia), escribimos las etiquetas para la categorización de la entrada y pulsamos *Publicar*. También podemos guardar la entrada como borrador para publicarla más adelante.

En cuanto a la sección *Ahora puedes*, ésta muestra de un simple vistazo el número de entradas, páginas, categorías, etiquetas y comentarios del sitio. Si pulsamos sobre cualquiera de sus nombres podemos pasar a su visualización y edición, si lo consideramos necesario. Los comentarios, además, aparecen clasificados en tres categorías: *Aprobados, Pendientes y Spam* (según nosotros mismos lo hayamos considerado). También es posible cambiar el tema de la bitácora rápidamente pulsando el botón *Cambiar tema* y eligiendo el que prefiramos de entre los disponibles.

Es muy importante también conocer la estructura de la barra superior disponible en las pantallas de la aplicación, que podemos ver en la figura 5-82.

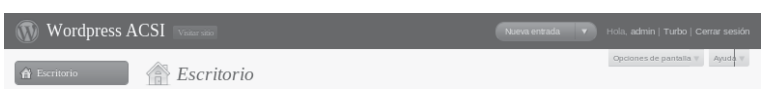


Figura 5-82. WordPress – Barra superior de WordPress

En esta barra, además de mostrar información sobre la página en la que nos encontramos actualmente, cambiar sus opciones de visualización o mostrar la ayuda e información sobre la sesión abierta, disponemos de un **menú desplegable** con diferentes opciones dependiendo de la página en la que nos encontramos, con las operaciones más importantes y comunes. Por ejemplo, para el *Escritorio*, podemos crear una nueva entrada,

crear una página, ver los comentarios o subir un nuevo fichero al sitio, como muestra la figura 5-83.

Figura 5-83. WordPress – Menú desplegable en la barra superior para el Escritorio

En la parte izquierda del *Escritorio* es donde encontramos el **menú de categorías administrativas**, que podemos ver en la siguiente figura 5-84.

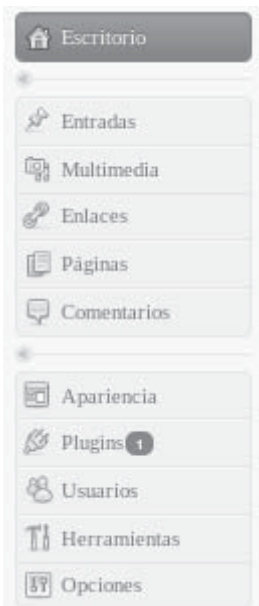


Figura 5-84. WordPress – Menú de categorías administrativas

La categoría con la que estamos trabajando aparece sombreada en este menú. A continuación comentaremos brevemente cada una de las opciones del menú:

- *Entradas.* Permite añadir nuevas entradas y editar las ya publicadas (o en borrador), las etiquetas asociadas a las entradas así como las diferentes categorías creadas para las mismas. Si hacemos clic en *Entradas* accedemos a un completo listado de todas las entradas de nuestro *blog*:

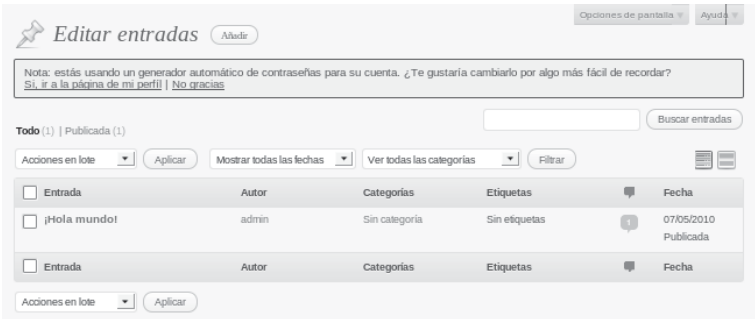


Figura 5-85. WordPress – Entradas

Podemos filtrar las entradas mostradas (publicadas o no, por fecha, por categoría), editarlas, moverlas a la papelera, etc. Para cada entrada podemos ver su título, el autor que la creó, la categoría a la que pertenece, las etiquetas que le han sido asignadas, el número de comentarios, la fecha de creación y su estado (publicado o no). Haciendo clic en el título de la entrada podemos pasar a su edición (véase la figura 5-86).

Figura 5-86. WordPress – Editar entrada

Al editar una entrada podemos ver en cualquier momento una vista previa de la misma, modificar su estado o cambiar el título y el contenido de ella. Para la edición del contenido contamos con un completo editor con las operaciones clásicas para la edición de texto e inserción de contenido multimedia (imágenes, vídeos embebidos, audio, o cualquier otro objeto), teniendo disponible también la vista HTML del contenido, si preferimos editarlo de este modo mejor.

En la ventana *Editar entrada* podemos elegir las etiquetas y categoría para clasificar nuestra entrada, escribir un pequeño resumen de la entrada, y otras opciones adicionales, como la posibilidad de permitir o no los comentarios y *pingbacks/trackbacks* a la página (enlaces externos desde otras páginas al contenido publicado).

- *Multimedia.* En esta categoría podemos ver el contenido de nuestra *Librería multimedia* y, si lo deseamos, añadir nuevos elementos. Por *Librería multimedia* entendemos el repositorio de imágenes, audio, vídeos… y cualquier otro objeto que hayamos subido al sitio para su inclusión en las entradas.
- *Enlaces.* Desde *Enlaces* podemos manipular los enlaces mostrados en el sitio, como referencias a otros *blogs*, URL de interés, *webs amigas*, etc. Al igual que para las entradas, disponemos de un listado de los enlaces con diversa información para cada uno, incluyendo una posible valoración propia y su estado.
- *Páginas.* Permite añadir y editar páginas adicionales a la principal en la que son mostradas las entradas. Es interesante al menos crear una página con información descriptiva del *blog*, por ejemplo. Por defecto es incluida la página *Acerca de*.
- *Comentarios.* Haciendo clic en *Comentarios* accedemos a un listado de los comentarios que han realizado los lectores a las entradas publicadas (ver la figura 5-87). Para cada comentario podemos ver el nombre y URL (si se incluyó) del autor, la fecha y contenido, y en respuesta a qué entrada fue realizado.

Figura 5-87. WordPress – Editar comentarios

Los comentarios pueden ser marcados como *aprobados, rechazados, spam*, o enviarlos a la papelera. Siguiendo esta clasificación de los comentarios también podemos filtrar los mostrados en la lista de la ventana *Editar comentarios*.

- ***Apariencia.*** A través de *Apariencia* podemos manipular los diferentes elementos que definen la parte visual del sitio WordPress: temas, *widgets*, el editor o la cabecera. Es posible cambiar el tema actual por alguno de los disponibles por defecto, o si lo necesitamos, instalar nuevos temas.

  Los *widgets* son objetos o pequeñas aplicaciones que pueden ser usadas en algunas zonas de nuestra bitácora. Por ejemplo, existen *widgets* para mostrar un calendario, insertar un formulario de búsqueda para las entradas del *blog*, mostrar *feeds RSS* de otras fuentes, una lista de enlaces o últimos comentarios. Para añadir *widgets* a las diferentes zonas de la página del *blog* basta con arrastrarlos a ellas, mostradas en la página *Widgets*, como muestra la figura 5-88, en la que ha sido añadido un formulario de búsqueda para la *Barra lateral 1*.

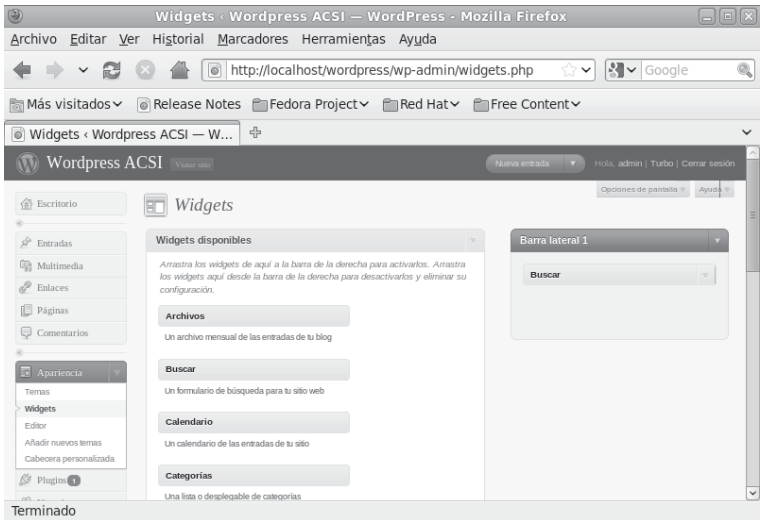


Figura 5-88. WordPress – Widgets

Una vez que el *widget* ha sido añadido, podemos editar su configuración, administrar los elementos que contienen, su aspecto, etc.

Otra de las páginas de gran importancia en la categoría *Apariencia* es *Editor de temas* (véase la figura 5-89). Con el editor tenemos la posibilidad de caracterizar al completo la visualización de los temas y plantillas utilizadas. Para cada elemento plantilla del tema es mostrado su código fuente, además podemos ver para el tema la *hoja de estilos en cascada (CSS)* asociada que contiene el formato y estilo a aplicar para cada uno de los elementos HTML mostrados.

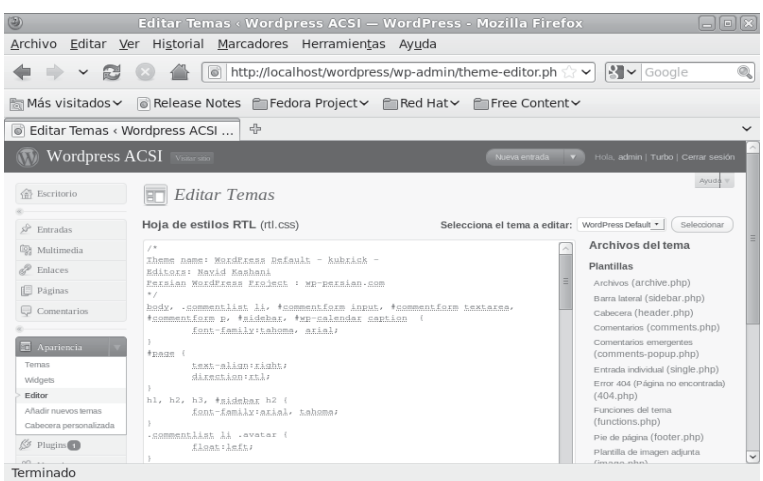


Figura 5-89. WordPress – Editar temas

Esto proporciona una gran flexibilidad al administrador del sitio WordPress, permitiendo editar el aspecto visual del *blog* con total libertad, sin ninguna limitación.

Podemos añadir nuevos temas al sitio, así como editar la cabecera haciendo clic en *Cabecera personalizada*, pudiendo editar el color de fuente, color superior e inferior, etc. En cambio, para cambiar la imagen de la cabecera por ejemplo, tras subir el fichero copiándolo en el directorio */var/www/html/wordpress/wp-content/themes/default/images* (directorio para imágenes del tema por defecto), hacemos uso del *Editor* sobre la *hoja de estilos CSS (style.css)* del tema, buscamos la sección *header* y en *background* escribimos la URL a la nueva imagen (figura 5-90).

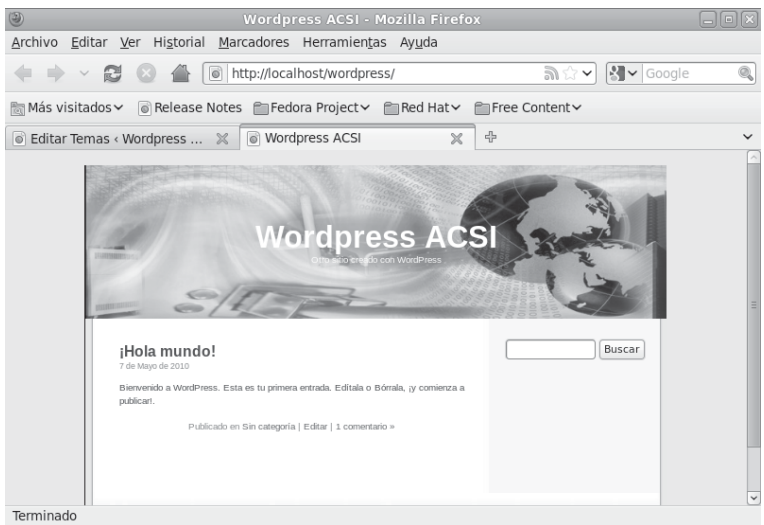


Figura 5-90. WordPress – Nueva imagen en la cabecera

- *Plugins.* Permite administrar los *plugins* añadidos al sitio WordPress. Podemos buscar nuevos *plugins,* activar o desactivar los instalados anteriormente, eliminarlos… Si una actualización se encuentra disponible para un *plugin* se nos notifica, en la propia ventana *Administrar plugins* y junto al enlace *Plugins* (mediante un número que indica cuántas actualizaciones pueden ser aplicadas).
- *Usuarios.* Permite ver y editar los usuarios y autores del sitio, además de poder cambiar y cumplimentar nuestro perfil como usuario *admin* (activar el editor visual al escribir, esquema de color usado, datos personales, información de contacto y otros datos, contraseña, etc.). En el listado de usuarios facilitado en la página *Usuarios*, para cada usuario se muestra el nombre de usuario, el nombre, la dirección de correo electrónico, su perfil y el número de entradas creadas (ver figura 5-91).

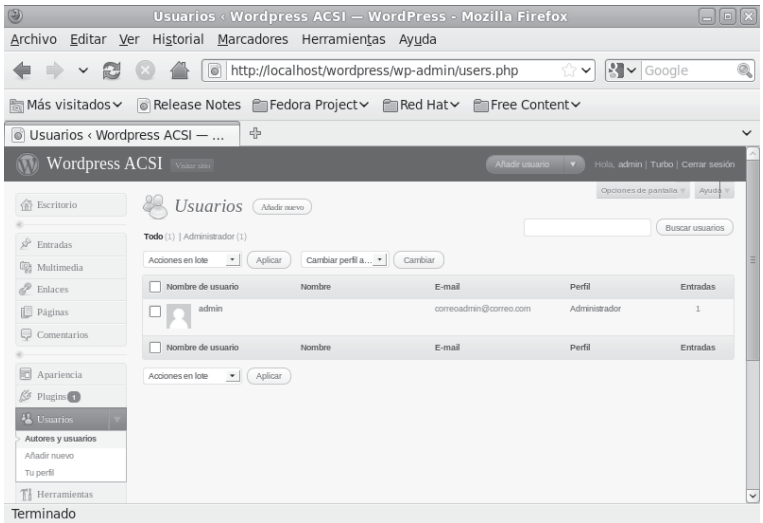


Figura 5-91. WordPress – Usuarios

Si queremos añadir un usuario nuevo pulsamos sobre *Añadir nuevo*; en la ventana *Añadir nuevo usuario* (véase la figura 5-92) debemos escribir los datos para la nueva cuenta de usuario (nombre de usuario, nombre, apellidos, email, dirección web, contraseña…). Podemos comprobar la fortaleza de la contraseña introducida y comunicarla inmediatamente al usuario por correo electrónico. Finalmente hay que elegir un perfil para el usuario de entre los siguientes: *Suscriptor, Administrador, Editor, Autor, Colaborador*.

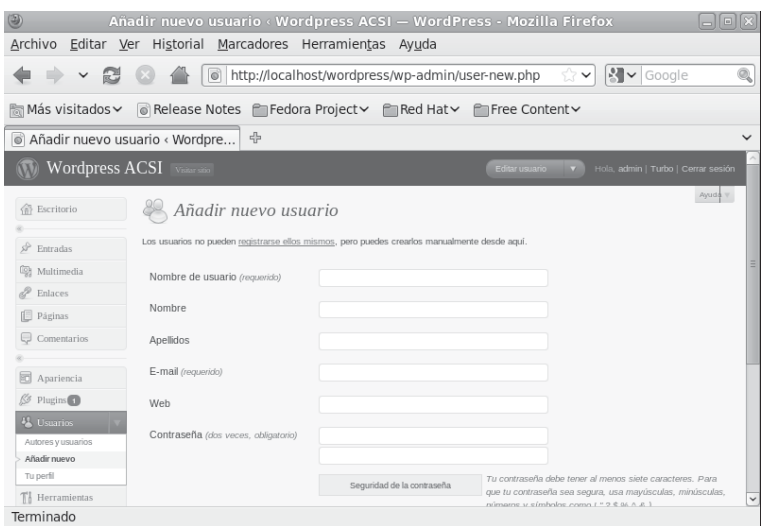


Figura 5-92. WordPress – Añadir nuevo usuario

- *Herramientas.* En *Herramientas* tenemos acceso a diferentes utilidades complementarias a WordPress y que pueden ser utilizadas para la mejora de su funcionalidad y rendimiento. Podemos importar nuevas, exportar las que tenemos instaladas a otro sistema, o actualizarlas.
- *Opciones.* La última categoría es *Opciones*. Dispone de diversas subcategorías que permiten establecer la configuración general del sitio (título, descripción, URL, email, zona horaria, opciones de fecha y hora…), opciones de escritura (acciones a tomar cuando se realiza una publicación) y lectura (por ejemplo, el número de entradas a mostrar), comentarios (habilitarlos, qué notificaciones se van a generar), multimedia, privacidad del sitio o los enlaces permanentes (tipos de enlaces a mostrar para las páginas y entradas del *blog*). En la figura 5-93 podemos ver las opciones para comentarios, por ejemplo.

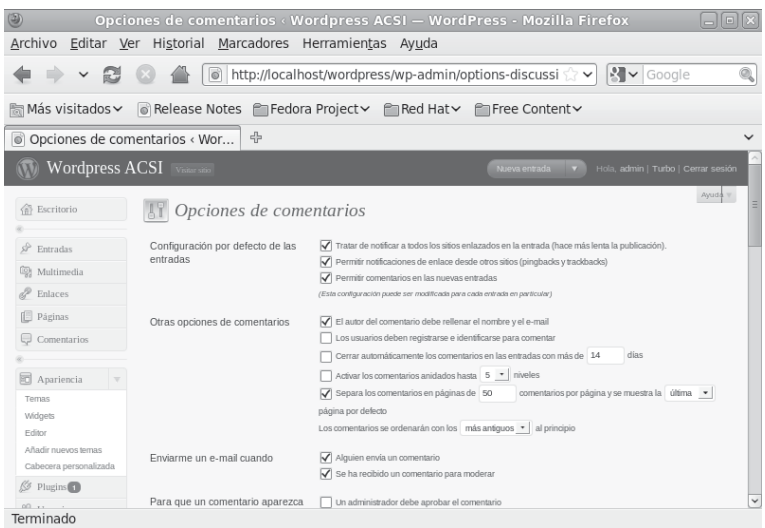


Figura 5-93. WordPress – Opciones de comentarios

Capítulo 6

POSICIONAMIENTO WEB

Internet ha tenido un impacto directo en los modos de trabajo, en el ocio, en el conocimiento, etc. Ha hecho surgir nuevos modelos de negocio y hoy en día las empresas se toman muy en serio su presencia en Internet.

De la misma manera que una tienda de un centro comercial intenta atraer a los clientes para que éstos visiten sus instalaciones y realicen compras, los dueños de sitios Web intentan atraer el máximo número de internautas a sus sitios para generar ingresos.

Se podría comparar el posicionamiento Web con la ventaja que puede tener un comerciante si sitúa su tienda en una de las calles más concurridas de la ciudad. Cuanta más gente pase por delante del escaparate, más posibilidades hay de que entren a comprar. Y no cabe duda de que uno de los medios principales a través del cual se puede hacer llegar tráfico a mi sitio Web, es a través de un buscador.

A continuación vamos a ver primero los conceptos básicos necesarios para entender mejor cómo se realiza el posicionamiento web y posteriormente, veremos cómo realizar un mejor posicionamiento de nuestro sitio Web.

6.1 CONCEPTOS BÁSICOS

Antes de empezar a utilizar las herramientas para mejorar y visualizar el posicionamiento de nuestro sitio web es necesario ver los conceptos básicos que nos permiten entender el funcionamiento de los buscadores.

6.1.1 ¿Qué es un buscador?

Un buscador es un sitio web o aplicación que partiendo de una cadena de texto, formada por las palabras clave que forman la búsqueda devuelve como resultado un conjunto de páginas web o documentos ordenados por importancia (*page rank*).

Los buscadores normalmente los podemos encuadrar en dos categorías: buscadores temáticos (cada página es indexada en función de una temática o categoría) y buscadores basados en motores de búsqueda (indexan las páginas en función de unas palabras clave).

Los buscadores utilizan robots, llamados también spiders o crawlers, que son programas que navegan automáticamente por la red, de enlace en enlace, de página en página procesando toda la información que van encontrando e indexando toda la información para poder utilizarla en las búsquedas.

El secreto de los buscadores está en el algoritmo de búsqueda o SEO (Search Engine Optimization) que define exactamente cómo funcionan esas arañas o robots y cómo procesan y valoran la información.

En la tabla 6-1 se muestran los buscadores más utilizados actualmente y su cuota de mercado. Sin duda alguna el buscador más utilizado y con mayor cuota de mercado es **Google** (www.google.es).

Tabla 6-1. Top 10 de los buscadores más utilizados

Rank	Buscador	Cuota de mercado
1	Google Search	65,4%
2	Yahoo! Search	15,3%
3	MSN/Windows Live/Bing Search	10,7%
4	AOL Search	2,8%
5	Ask.com Search	1,8%
6	My Web Search Search	1,0%
7	Comcast Search	0,5%
8	NexTag Search	0,3%
9	BizRate Search	0,3%
10	Yellow Pages Search	0,3%

Fuente http://blog.nielsen.com/nielsenwire/online_mobile/top-u-s-online-search-providers-november-2009/

¿Sabías que...?

Google (www.google.es) fue fundado en septiembre de 1998 por dos estudiantes de doctorado de la Universidad de Stanford, Larry Page y Sergey Brin.
Desde ese momento Google ha marcado un antes y un después en lo que a herramientas de búsqueda se refiere y es el punto de referencia para el resto de buscadores.
Su éxito se basa en la importancia que Google da a los enlaces entrantes que tiene una página. Google entiende que si dos páginas compiten por aparecer en una búsqueda, será más importante la que más enlaces a ella tenga desde otras páginas. Si muchas páginas hablan de otra página será por algo ¿no?

6.1.2 ¿Qué es una palabra clave?

Las palabras clave o keywords permiten identificar un sitio web. Los robots de los buscadores obtienen las palabras clave de una web a partir de diferentes fuentes y las van clasificando según sus palabras clave. Los robots pueden obtener las palabras clave de nuestro sitio automáticamente a través del contenido y los enlaces de nuestras páginas web o podemos ayudarles nosotros a que nos identifiquen correctamente escribiendo nosotros las palabras clave en la cabecera de la página web (utilizando la etiqueta *metadata*) o realizando un mapa (*sitemap*) de nuestro sitio web.

Dado que el proceso de búsqueda se realiza a través de las palabras clave, es muy importante elegir muy bien las palabras clave que van a identificar el sitio web. Si utilizamos palabras clave muy genéricas entonces "competiremos" con otros muchos sitios web para obtener un posicionamiento en el buscador. Por otro lado, si utilizamos palabras clave muy específicas tendremos menos "competidores" pero será más difícil nuestra localización.

Las palabras clave las podemos clasificar en las siguientes categorías:

- **Palabras Clave Genéricas (Core Keyword).** Son las palabras clave más populares de nuestro sitio web, directamente relacionadas con la actividad del sitio web. Por ejemplo, si el sitio web está destinado a la venta de coches la palabra clave genérica será "venta de coches".
- **Palabras Clave Secundarias.** Son grupos de palabras clave directamente relacionadas con las palabras clave genéricas. En nuestro ejemplo anterior, las palabras clave secundarias serían "venta coches bmw", "venta de coches audi", "venta de coches seat", etc. Estas palabras aparecerían en los primeros subniveles de nuestra web.
- **Palabras Clave de Long Tail.** Son palabras clave secundarias y destinadas a aparecer en páginas que se encuentren a 2 o 3 clics de la principal. Siguiendo con el ejemplo las palabras clave Long Tail serían: "venta bmw 320", "venta ford fiesta", etc. Como se verá más adelante, estas palabras clave tienen un alto grado de importancia.

Según la "Teoría del Long Tail" la mitad de las visitas que recibe un sitio web se deben a las búsquedas relacionadas con palabras clave genéricas y secundarias y la otra mitad se producen debido a las palabras que forman parte del Long Tail. Por tanto es muy importante definir y colocar correctamente las palabras clave en cada una de las páginas del sitio web.

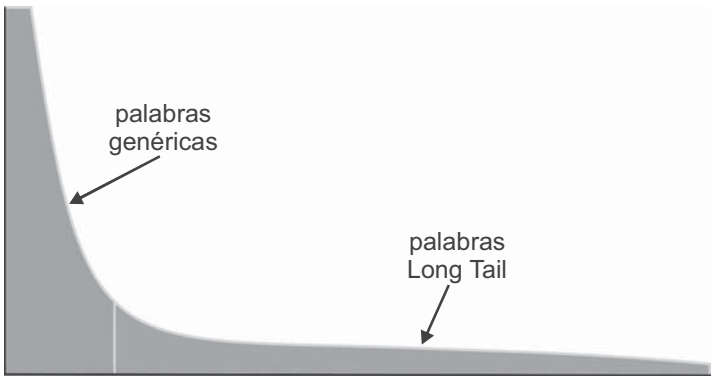


Figura 6-1. Importancia de las palabras clave en las búsquedas

6.1.3 Page rank

El *Page Rank* es un concepto introducido por Google, utilizado para asignar numéricamente la importancia a una página web con respecto a otra, y por ello la página de mayor importancia aparecerá primero en el resultado de la búsqueda.

El algoritmo que asigna el *Page Rank* de Google, no sólo cuenta el número de enlaces de otras páginas a la que analiza como se venía haciendo anteriormente, sino que le asigna un mayor o menor peso dependiendo de la importancia de la página que se enlaza ya que no es lo mismo un enlace a nuestra página desde una página personal o desde una página de una empresa o institución.

6.2 POSICIONAMIENTO DE UNA WEB

Ahora que conocemos cómo funciona un buscador, se puede hacer el camino más sencillo a sus robots para que indexen mejor nuestras páginas. Podemos decir entonces que a la hora de diseñar una página Web hay que tener en cuenta tanto a nuestros usuarios como a los robots de los buscadores que también nos van a visitar.

6.2.1 Dar de alta la web en los buscadores

Existen dos formas para dar de alta nuestra página en los buscadores:

- **Automáticamente.** Como hemos comentado antes, los robots de los buscadores van analizando e indexando toda la web a partir de los enlaces que tienen las diferentes páginas. Por lo tanto, la forma natural de dar de alta nuestra web es que el buscador nos localice y nos dé de alta a partir del enlace de otra web.
- **Manualmente.** También es posible dar de alta nuestra web a partir de los formularios que ofrecen los diferentes buscadores. En la tabla 6-2 podemos ver la dirección, para dar de alta nuestra web, en los buscadores más utilizados.

Tabla 6-2. URL para el alta en buscadores

Buscador	URL
Google	http://www.google.com/intl/es/addurl.html
Yahoo!	http://siteexplorer.search.yahoo.com/es/free/submit (es necesario tener una cuenta en Yahoo!)
Bing	http://www.bing.com/webmaster/SubmitSitePage.aspx

El tiempo que tarda en aparecer en los buscadores la Web depende de muchos factores. Si se utiliza el alta manual, el tiempo de aparición puede variar dependiendo de la cola de procesos pendientes que tenga el buscador y de la velocidad con la que actualice sus índices de búsqueda. Y si se utiliza el método natural, ahora todo dependerá de la frecuencia de rastreo de las arañas del buscador en la Web donde aparece el enlace a la propia y de nuevo la velocidad con la que actualicen los índices del buscador.

6.2.2 Indexación de la web

Para conocer cómo los robots ven nuestra web podemos utilizar herramientas simuladores de spiders como http://webconfs.com/search-engine-spider-simulator.php. Si realiza una revisión de una página, esta herramienta muestra todo el texto que puede encontrar en ella, los enlaces disponibles desde la misma y los metadatos de ésta.

A modo de ejemplo, en la figura 6-2 podemos ver la visión de un spider de la web www.ual.es.

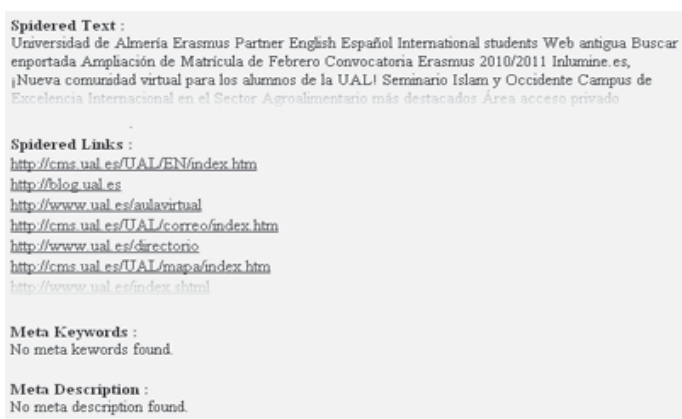


Figura 6-2.Visión de la web www.ual.es por un spider

Además si lo deseamos, Google ofrece la posibilidad de ver la última vez que indexó el robot de Google una página y sus contenidos. Para ello, en el buscador de Google (www.google.es) buscamos la página web, pulsamos el enlace *En caché*. Tal y como se muestra en la figura 6-3, en el recuadro gris que aparece en la parte superior podemos ver la fecha de la última vez que el robot de Google analizó la web.

Figura 6-3. Caché de Google de la web www.ual.es

Resulta interesante comprobar la última fecha de actualización de la web que estamos analizando para detectar posibles errores de indexación. Además, si nos fijamos bien tanto en el ejemplo del spider como en la caché de Google lo único que se indexa es texto.

Normalmente los robots llegan a la página principal del sitio y de ahí van siguiendo los enlaces que van encontrando sucesivamente, si alguna Web no se encuentra correctamente enlazada, el robot no podrá encontrarla. Las limitaciones de los robots para procesar el contenido de una web son:

- No procesan elementos multimedia (imágenes, sonido, video) ya que sólo procesan texto.
- No trabajan bien con los formularios por lo que, lo normal, es que no puedan acceder a las páginas a las que se acceda a través de un formulario.
- No procesan código javascript.
- No indexan contenido en flash.
- No procesan páginas con marcos o iframes.

Para poder obtener un mejor posicionamiento de nuestra web es importante conocer el orden de prioridad para los buscadores:

1. **URL de la página.**
2. **El título o etiqueta <title>.** La etiqueta *title* de cada archivo debe ir acorde a su contenido. El número máximo de caracteres para la etiqueta *title* es de 160, el resto de caracteres son descartados por el buscador.

3. **Los metadatos <keywords><description>.** El número aconsejado de *keyword*s por página no debe superar las 25–30 y por otro lado el número de caracteres para *description* tiene un tope de 160, el resto son descartados por el buscador.

4. **El contenido de la página.** El orden de importancia que se le otorga al contenido de la página (body), también tiene su orden donde en primera posición tenemos los títulos *<h1><h2><h3>* y luego tenemos las palabras resaltadas con negrita *<b>* o **.

Nota

Abusar del uso de alguna etiqueta es considerado una mala práctica de posicionamiento y por tanto penalizado por los buscadores.

6.2.3 Elección de palabras clave

Para seleccionar las mejores palabras clave para nuestro sitio es recomendable que utilicemos la herramienta Keyword Tool de Google (https://adwords.google.es/select/KeywordToolExternal).

A través de *Keyword Tool* (véase la figura 6-4) podemos ir escribiendo las diferentes palabras clave, su índice de competencia y las búsquedas mensuales que generan. De esta forma podemos "afinar" las palabras clave de nuestro sitio web.

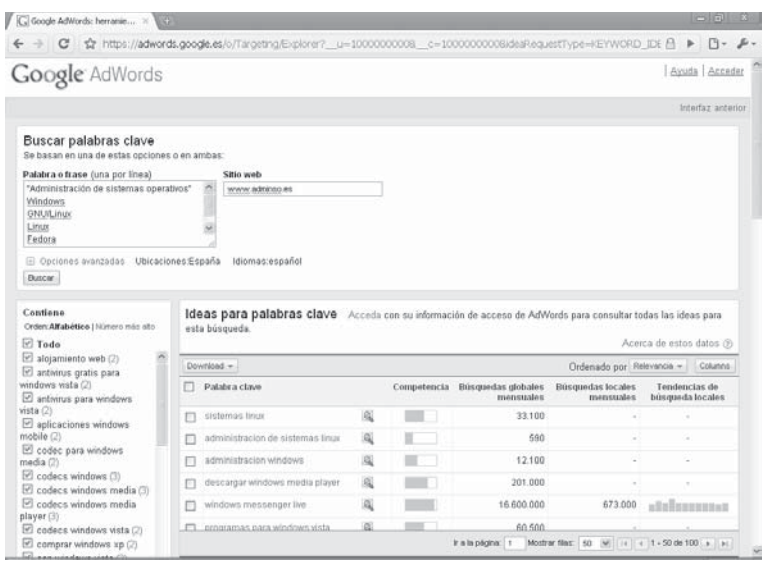


Figura 6-4. Google AdWord – Elección de palabras clave

Nota

Intentar posicionar una web con muchas palabras es algo complejo pero en este caso son combinaciones de las palabras.

6.2.4 Popularidad de una web

Para medir los enlaces que hay en otras páginas hacia tu sitio debes utilizar los propios buscadores. En muchas ocasiones tienen una herramienta de búsqueda avanzada o alguna ayuda que explica cómo puedes hacer esta búsqueda. Por ejemplo, Google incluye esta opción y para utilizarla deberemos buscar por link, seguido por la dirección de la que quieres contar el número de enlaces, de esta forma:

```
link:http://www.tudominio.com
```

Figura 6-5. Enlaces al sitio www.ual.es

La manera en la que Google denomina a la popularidad web es **PageRank**. En una escala de 1 a 10. Y éste va pasando de páginas a páginas a través de los enlaces. Cuanto más PageRank tengan las webs que enlazan tu sitio mayor será el tuyo.

Puedes comprobar el PageRank de una web utilizando las herramientas para webmaster de Google, la barra de Google para los buscadores o con páginas como ésta: http://www.mipagerank.com.

6.2.5 Buenas prácticas para mejorar la indexación

Para ayudar a que los buscadores nos indexen correctamente es importante escribir correctamente las palabras clave de nuestro sitio y facilitarle al buscador el mapa de nuestro sitio web.

6.2.5.1 PALABRAS CLAVE

A la hora de generar las diferentes páginas web debemos tener cuidado a la hora de escribir el título, la descripción del sitio y las palabras clave que nos identifican. A continuación vamos a ver un ejemplo:

```
<HEAD>
      <TITLE> Curso de portales Web </TITLE>
      <META name="description" content="Curso de diseño y creación
de portales web">
      <META name="keywords" content="web html css javascript php
mysql LAMP moodle">
</HEAD>
```

6.2.5.2 MAPA DEL SITIO O SITEMAP

Un sitemap es un archivo XML que contiene una lista de las páginas del sitio junto con información adicional sobre éstas (por ejemplo frecuencia de actualización de los contenidos, última actualización). Este archivo es consultado por los robots de los buscadores para así poder facilitarle la indexación de nuestro sitio web.

Para que este archivo XML sea comprendido por los robots de los buscadores debemos generarlo siguiendo las pautas de su estándar:

Contenidos obligatorios del sitemap

El protocolo sitemap se construye en base a etiquetas XML (Tags) incluidas en un archivo con codificación UTF-8.

Nota

Los valores de datos (por contraposición a las etiquetas) deben utilizar códigos de escape para ciertos caracteres especiales, tal como se acostumbra en HTML.
Por ejemplo, las comillas dobles (") deben ser reemplazadas por " y los signos menor (<) y mayor (>) por &lt y &gt, respectivamente.

A continuación podemos ver un ejemplo de sitemap donde nuestra web está formada por dos únicas páginas HTML. La página de inicio o index que se encuentra en http://www.misitio.com y una de contacto en http://www.misitio.com/contacto.htm.

```
<?xml version="1.0" encoding="UTF-8"?>
<urlset xmlns="http://www.sitemaps.org/schemas/sitemap/0.9">
  <url>
      <loc>http://www.misitio.com/</loc>
  </url>
  <url>
      <loc>http://www.misitio.com/contacto.htm</loc>
  </url>
</urlset>
```

Cada nodo <url> padre puede contener (además del nodo <loc> hijo obligatorio) cierta información adicional útil para que el proceso de indexación se realice más inteligentemente. Los nodos opcionales, que son importantes, son:

- **<lastmod>Fecha</lastmod>.** Se refiere a la fecha de la última modificación de la página. Esta fecha debe expresarse en formado AAAA-MM-DD, por lo que el 18 de abril de 2010 será 2010-04-18.

- **<changefrec>Frec</changefrec>.** Se refiere a la frecuencia con la que cambia el contenido de la página que figura en <loc> y será un dato que indicará a la araña con qué frecuencia volver a visitar el sitio. Es un valor orientativo, por lo que no quiere decir que deba cumplirse forzosamente. Si al robot le indicamos que acceda diariamente y luego el contenido apenas sufre cambios, el buscador tomará sus propias decisiones de frecuencia.

  La variable *Frec* puede tomar alguno de los siguientes valores: *always* (siempre, para páginas que cambian cada vez que se muestran, típicamente, las dinámicas), *hourly* (a cada hora), *daily* (diariamente), *weekly* (semanalmente), *monthly* (mensualmente), *yearly* (anualmente) o *never* (nunca, muy usado para páginas archivadas).

- **<priority>Valor</priority>.** Permite indicar la importancia que tiene la página que figura en <loc> respecto de las demás que componen el sitio. Es simplemente una manera de indicar prioridades relativas dentro del sitio. Los posibles valores son entre 0 y 1. El valor por defecto es 0,5.

Volviendo al ejemplo anterior y agregando los atributos opcionales:

```
<?xml version="1.0" encoding="UTF-8"?>
<urlset xmlns="http://www.sitemaps.org/schemas/sitemap/0.9">
  <url>
       <loc>http://www.misitio.com/</loc>
       <lastmod>2010-04-20</lastmod>
       <changefrec>monthly</changefrec>
       <priority>0.8</priority>
  </url>
  <url>
       <loc>http://www.misitio.com/contacto.htm</loc>
       <lastmod>2010-02-20</lastmod>
       <changefrec>yearly</changefrec>
       <priority>0.3</priority>
  </url>
</urlset>
```

El archivo XML se aloja en el servidor como un archivo más del sitio, con la salvedad de que puede contener las URL contenidas en el mismo directorio en que se encuentra o en otros contenidos en él, nunca al revés. Por ejemplo, si ubicamos el archivo *sitemap.xml* en el directorio *ejemplo.com/catalogo/* no puede incluir URL que se encuentren en *ejemplo.com*.

Por esta razón es aconsejable guardar el fichero *sitemap.xml* directamente en el directorio principal del sitio y desde éste listar todos los archivos.

Por último, es recomendable indicar a los buscadores la existencia del mapa del sitio. Por ejemplo, podemos utilizar la *Herramienta para webmaster* de Google para enviarle el mapa de nuestro sitio.

6.2.5.3 ROBOTS.TXT

Como hemos comentado anteriormente, los robots o spiders de los buscadores van analizando y clasificando todo el contenido de nuestro sitio web. Es posible que tengamos carpetas con información innecesaria para el contenido de búsqueda (p.e. una carpeta de imágenes, una zona de la web privada) y por lo tanto queramos excluir dichas carpetas para que no las procesen los robots de los buscadores.

Para establecer las exclusiones de nuestro sitio web utilizamos el protocolo de exclusión de robots. Para ello debemos generar el fichero *robots.txt.*

A continuación podemos ver varios ejemplos del fichero *robots.txt.*

- Permite que todos los robots visiten todos los archivos que se encuentran almacenados en el código raíz de la página web porque el comodín "*" especifica todos los robots:

```
User-agent: *
Disallow:
```

- Niega el acceso a todos los robots en todos los archivos almacenados en el directorio raíz (/):

```
User-agent: *
Disallow: /
```

- En el siguiente ejemplo se excluyen para todos los buscadores los directorios */cgi-bin, /images* y */tmp.*

```
User-agent: *
Disallow: /cgi-bin/
Disallow: /images/
Disallow: /tmp/
```

6.2.6 Cómo detectar y resolver problemas de indexabilidad en la práctica

En ocasiones es posible que existan problemas de indexación en nuestras páginas Web. El indicador más claro de que un buscador no está indexando correctamente tu Web es consultar el número de páginas que tiene indexadas. Si este número no se corresponde con el número de páginas que contiene tu sitio Web, eso significa que hay un problema en la indexación.

Para ver la indexación de un sitio web puede utilizar herramientas para webmaster de Google que nos muestra el número de páginas que Google tiene indexadas o consultar en el buscador:

```
site:http://www.nuestrositio.com
```

Figura 6-6. Resultado de la búsqueda site: http://www.ual.es

Tal y como puede ver en la figura 6-6, el buscador nos devuelve el número de páginas que tiene indexadas el buscador. Si éstas se corresponden con el número de archivos HTML que tenemos subidos a nuestro servidor es que todo va correcto, si el número es excesivamente menor, tenemos un problema.

En este ejemplo en particular, tratándose de sitio web tan extenso, comprobar si la cifra de 6.900 páginas indexadas se corresponde con el número real de páginas del sitio sería algo complejo, pero los gestores de contenidos de páginas web como las de la UAL te indican que número de páginas contienen y comparándolo con el obtenido nos hacemos una idea de si hay problemas de indexación si el que aparece en Google fuera muy inferior al del gestor.

Nota

Normalmente el número de páginas indexadas que ofrece Google a través de este sistema no es del todo fiable y se debe utilizar como un número orientativo.

De todas maneras, los buscadores se toman su tiempo para indexar un sitio Web por lo que hay que ser constantes pero también pacientes consultando estos datos.

Si detectamos problemas de indexabilidad las causas pueden ser debidas a:

- La existencia de trabas, vistas anteriormente, del tipo flash, javascript, formularios, marcos o iframes.
- Revisar si existe uso de variables de sesión en URL ya que esto puede provocar que se pueda acceder al mismo contenido con distintas URL, lo cual puede llevar a confusión a los robots y no indexar correctamente el contenido.
- Un problema muy en auge es el de las listas negras de IP, donde se encuentran listadas todas las direcciones de servidores que realizan SPAM. Estos servidores están marcados por los buscadores y a veces no indexan sus contenidos. Por lo que es una práctica muy común comprobar si el hosting de mi web es compartido

con algún dominio que practique SPAM. Puede utilizar esta web para comprobar quiénes son nuestros "vecinos": http://www.myipneighbors.com/

Para saber más.

Blogs sobre posicionamiento:
http://www.seoprofesional.com
http://www.google.dirson.com
http://seoblog.es

6.2.7 Malas prácticas de posicionamiento

Normalmente, una página se convierte en popular por la calidad de su contenido, aunque existen estrategias o métodos que intentan buscar los puntos débiles o fallos de los buscadores para posicionar de forma irregular una web.

A continuación se describen algunos de estos métodos:

- **Cloacking o encubrimiento.** Consiste en mostrar a los robots un contenido distinto al de la web para que se indexe mejor.
- **Contenido duplicado.** Se basa en realizar copia parcial o absoluta de un sitio web con el fin de duplicar enlaces y palabras, consiguiendo una mejor indexación.
- **Agregar keywords o palabras clave sin relación al contenido**. Técnica muy común hace unos años, aunque carece de efectividad en la actualidad.
- **Ocultar texto.** Poner palabras clave en la página con el mismo color que el fondo o colocar letras extremadamente pequeñas al final del documento no sirve. Google las encuentra con demasiada facilidad y penaliza al sitio web.
- **Apilamiento en el título.** Consiste en poner más de una etiqueta *title*, la idea es poner muchos títulos para agregar más palabras clave.
- **Distribuir vírus, troyanos o malware.**

El uso de estas técnicas puede que en cualquier momento lleve a perder posiciones en el índice del buscador o incluso la eliminación.

6.3 HERRAMIENTAS PARA WEBMASTER

Existen multitud de herramientas que pueden resultar muy útiles en las tareas de posicionamiento web. Las más interesantes son aquéllas que permiten monitorizar los resultados de nuestro trabajo y muestran los posibles problemas de indexación de nuestros sitios web.

A continuación vamos a ver las herramientas más utilizadas actualmente entre los webmasters.

6.3.1 Google Analytics

Google Analytics (https://www.google.com/analytics) es una de las principales herramientas gratuitas que permite recopilar, ver y analizar datos sobre el tráfico de tu sitio web. Por ejemplo, se puede ver entre otras cosas, la ubicación geográfica de nuestros visitantes, qué navegador utilizan, qué resolución de pantalla, o qué criterios de búsqueda utilizan para encontrar nuestro sitio web. También ofrece la posibilidad de realizar comparaciones en el tiempo sobre las visitas y ubicaciones de usuarios.

Para acceder al servicio sólo es necesario tener una cuenta de Google y darse de alta en el servicio.

Figura 6-7. Registro en Google Analytics

Para registrarnos es necesario rellenar una serie de formularios con datos generales sobre el sitio web (país, zona horaria…) e información sobre nuestra cuenta.

Para terminar con el registro se debe aceptar el acuerdo de usuario. Tras este paso, se suministra el código de seguimiento en javascript que se debe incluir antes de la cabecera de las páginas que se van a analizar.

En la figura 6-8, se puede ver un ejemplo del código de seguimiento suministrado, así como unas pestañas con las que se pueden personalizar o seleccionar características avanzadas de análisis.

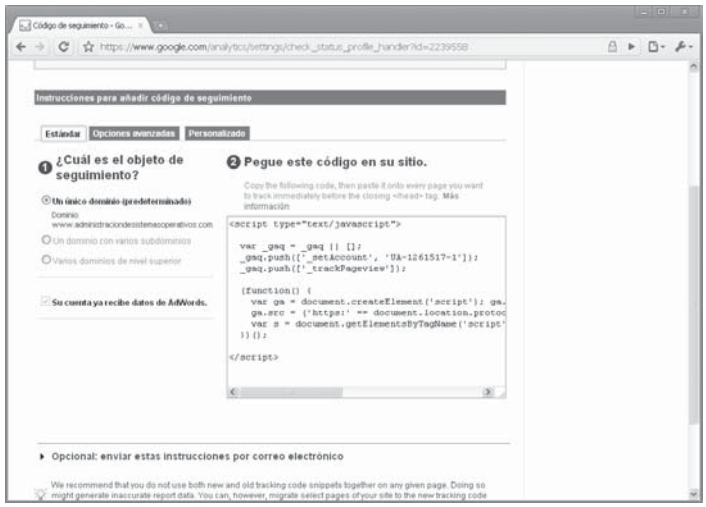


Figura 6-8. Código de seguimiento

Tras realizar el registro, Google Analytics comenzará a registrar información sobre las visitas recibidas en la web.

Nota

Las estadísticas de las cuentas de Google Analytics se actualizan cada 24 horas.

Una vez trascurridas al menos 24 horas del registro del sitio web, el sistema ya contará con suficiente información para poder generar informes. Tal y como podemos ver en la figura 6-9, si accedemos a la página principal de Google Analytics podremos ver el estado de los diferentes dominios que tengamos. Los datos más importantes que podemos ver son:

- Se realiza la correcta recepción de datos.
- El número de visitas.
- El tiempo medio de las visitas.
- El porcentaje de rebotes de las visitas.
- Y el porcentaje de mejora del número de visitas.

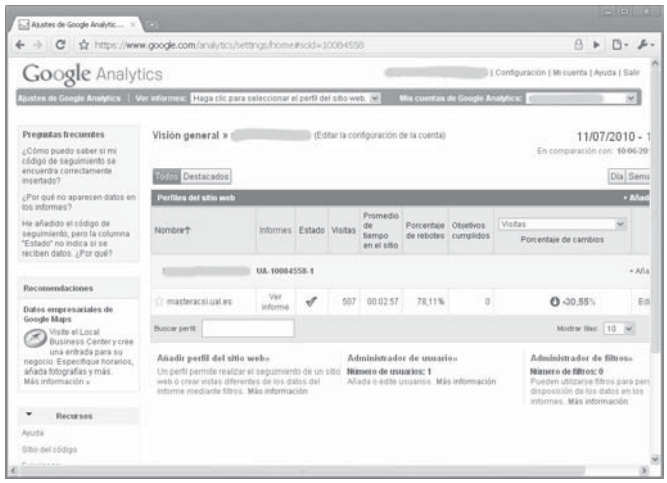


Figura 6-9. Inicio de Google Analytics

Si deseamos una información mucho más detallada podemos pulsar en el enlace *Ver Informe* para ver el informe del sitio web.

Figura 6-10. Google Analytics – Panel

Tal y como podemos ver en la figura 6-10, el informe queda dividido en dos zonas bien diferenciadas: *Menú de opciones* (donde se pueden seleccionar las distintas opciones del informe) y el *Visor de datos*.

A continuación veremos los informes más importantes que nos permite obtener Google Analytics:

- **Panel**. Muestra un resumen de las estadísticas más importantes del sitio, tal y como se puede apreciar en la figura 6-10. En el panel se muestra la evolución de las visitas del sitio, el uso del sitio, la visión general de los usuarios, el gráfico de ubicación de las visitas, una visión general de las fuentes de tráfico y una visión general del contenido.

  Dentro del panel podemos encontrar un resumen (véase la figura 6-11) con la siguiente información general del sitio:

  - **Visitas.** Es uno de los elementos más importantes y muestra el número total de visitas que ha tenido nuestro sitio web.
  - **Usuarios únicos.** Google Analytics diferencia entre *Visitas* y *Usuario único*. Un usuario único es un usuario que puede realizar varias visitas en nuestro sitio web. El número de visitas y usuarios de nuestro sitio web nos pueden proporcionar el nivel de fidelidad de su público. Cuanto más bajo sea el número de visitantes únicos en relación con el de visitas, más fiel es su público.
  - **Número de páginas vistas.** Es el total de páginas que han visto los usuarios en sus diferentes visitas. Este valor es muy importante ya que no sirve tener 10.000 visitas a mi web si no pasan de la página principal, es como si por delante del escaparate de mi tienda se asoman 10.000 personas pero nadie entra a la tienda.
  - **Promedio de páginas vistas.** Es el número de páginas que visualiza un usuario en una visita.
  - **Tiempo en el sitio.** Es el tiempo que cada visitante pasa en nuestra web. Este tiempo indica si le gusta y disfruta navegando por ella o al contrario, que visita la web pero se va pronto. Por regla general, cuanto más alto sea, mejor.
  - **Porcentaje de rebote y nuevas visitas.** Muestra la fidelidad de los usuarios en nuestro sitio web. Un alto porcentaje de rebote significa que los usuarios vuelven a acceder a nuestro sitio web.

- **Usuarios**. Muestra una información general de los usuarios y las visitas que se han ido realizando en el sitio web. Además, en el perfil técnico muestra información sobre los navegadores que han utilizado así como su velocidad de conexión. En la figura 6-12 podemos ver un ejemplo de cómo se muestra la información.

Figura 6-11. Google Analytics – Resumen

Figura 6-12. Google Analytics – Usuarios

Si pulsamos en el submenú *Gráfico de visitas por ubicación* podemos obtener el gráfico que podemos ver en la figura 6-13 donde se muestra un resumen de las visitas que hemos obtenido en los diferentes países.

Figura 6-13. Google Analytics – Gráfico de visitas por ubicación

Si pulsamos sobre un determinado país entonces podremos ver las visitas por ciudades (véase la figura 6-14).

Figura 6-14. Google Analytics – Gráfico de visitas por ciudad

- **Fuentes del tráfico**. Tal y como podemos ver en la figura 6-15, muestra información sobre la procedencia del tráfico recibido, por ejemplo: si el tráfico proviene directamente, a través de buscadores, de sitios web de referencia, etc. De esta forma, podemos conocer qué páginas nos enlazan y el volumen de visitas que nos aportan. Esto nos ayuda a centrar nuestros esfuerzos de promoción en aquellas webs que nos reportan un mejor resultado.

  Por otro lado, indica el porcentaje de visitas que traen los resultados orgánicos de los buscadores. Este dato a lo largo del tiempo, junto con el análisis de las palabras clave, es fundamental para evaluar el éxito de nuestra estrategia y ayudar a realizar los ajustes pertinentes.

- **Palabras clave.** El análisis de las palabras o expresiones clave que utilizan los usuarios para llegar al sitio web a través de los buscadores puede proporcionar información interesante sobre qué buscan los visitantes y por tanto qué contenido se debe mostrar. Tal y como podemos ver en la figura 6-16, en el listado de palabras clave podemos ver datos estadísticos sobre las palabras clave como el número de visitas que nos han reportado, páginas vistas, tiempo en el sitio, etc.

Figura 6-15. Google Analytics – Fuentes de tráfico

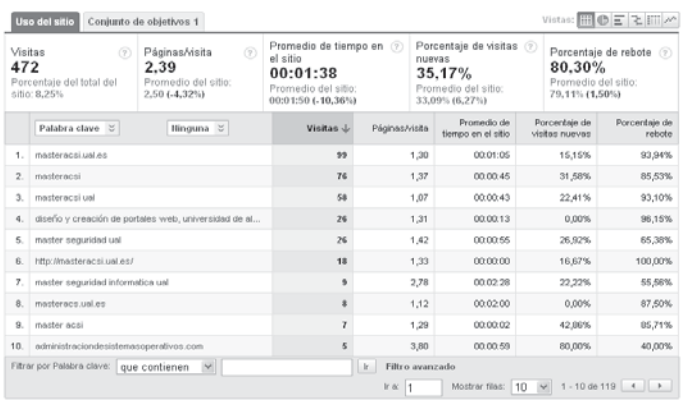


Figura 6-16. Google Analytics – Palabras clave

- **Contenido**. Muestra las estadísticas sobre los contenidos más visitados del sitio, tiempo dedicado por visitas…

    - **Páginas más populares.** En la sección *Contenido general* dentro del menú *Contenido* podemos encontrar un análisis de las páginas más visitadas de nuestro sitio web (véase la figura 6-17). Para cada página podemos ver su

número de visitas, promedio de tiempo en la página, porcentaje de rebote, etc. Este informe es muy útil para conocer las páginas que le gustan más a los usuarios.

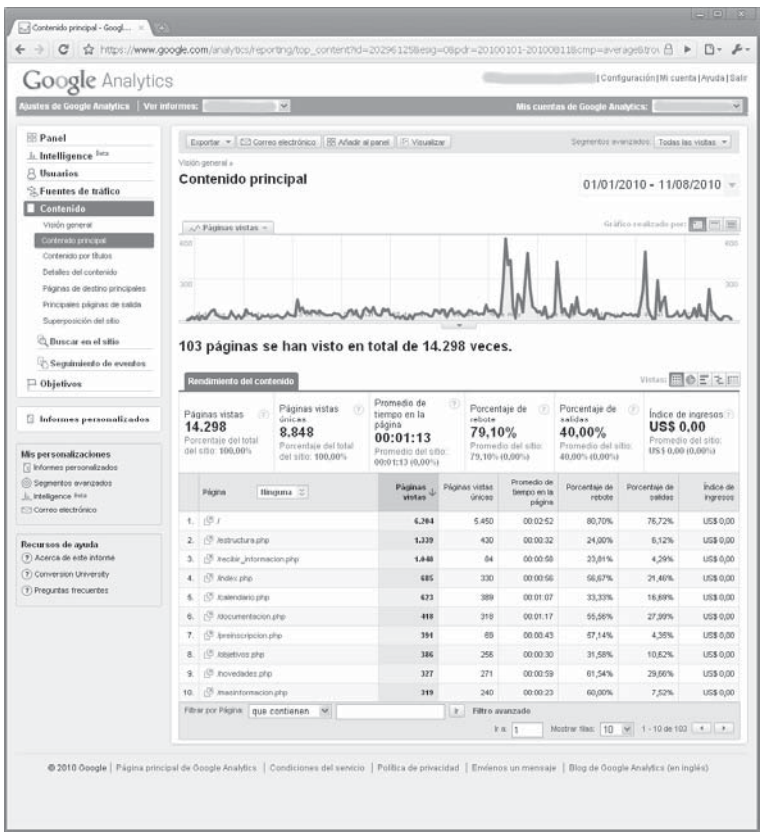


Figura 6-17. Google Analytics – Páginas más populares

- **Objetivos.** Se utilizan para definir los objetivos (páginas web) a los que pretendemos que accedan los usuarios (p.e. la web para la compra de un producto).

Como se puede ver, Google Analytics presenta gran cantidad de información y la mejor forma de aprender es navegar por sus diferentes informes.

6.3.2 Herramientas para webmaster de Google

Las herramientas para webmaster de Google (https://www.google.com/webmasters/tools/) ofrecen datos basados en las visitas a un sitio web y en los enlaces que apuntan a éste. Google ofrece mecanismos para detectar posibles errores de indexación

por parte de sus robots, un listado de posibles errores en la estructura de nuestra web y la posibilidad de cargar sitemaps de nuestras páginas.

Para hacer uso de esta herramienta es necesario, al igual que en Analytics, disponer de una cuenta de usuario de Google con la cual podemos acceder al sistema.

Una vez dentro agregar nuevos sitios es muy sencillo. Pulsamos sobre el botón *Añadir un sitio* (ver figura 6-18) e introducimos la dirección URL del mismo.

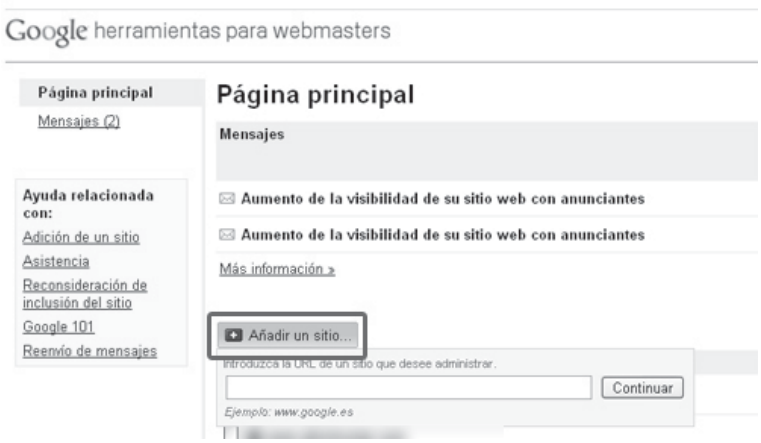


Figura 6-18. Herramientas para webmaster – Nuevo sitio

Una vez dado de alta el sitio debemos demostrar la propiedad del sitio (véase la figura 6-19). Para ello Google nos permite varias formas: añadiendo una etiqueta *meta etiqueta* en la página principal, subir un archivo HTML, o añadiendo un registro DNS. Realizamos la opción que más nos interese y pulsamos el botón *Verificar* para *verificar* la propiedad de nuestro sitio.

Una vez verificada la propiedad del sitio, tal y como muestra la figura 6-20, podemos obtener la siguiente información referente a nuestra web:

- **Consultas de búsquedas con impresiones y clics.** Esta información es muy útil ya que podemos ver cómo nos han encontrado en el buscador los usuarios.
- **Enlaces que existen al sitio web.**
- **Errores de rastreo.** Nos permiten ver los errores de rastreo que han detectado los robots de búsqueda de Google. Los errores más frecuentes suelen ser de enlaces inaccesibles. Resulta importante solucionar estos errores por lo que deberemos pulsar en cada categoría para ver cada uno de los errores localizados.
- **Palabras clave encontradas.** En esta sección podemos ver las palabras clave que identifican nuestro sitio web así como su importancia.
- **Sitemaps.** Permite indicarle a Google la ubicación de los mapas de nuestro sitio web.

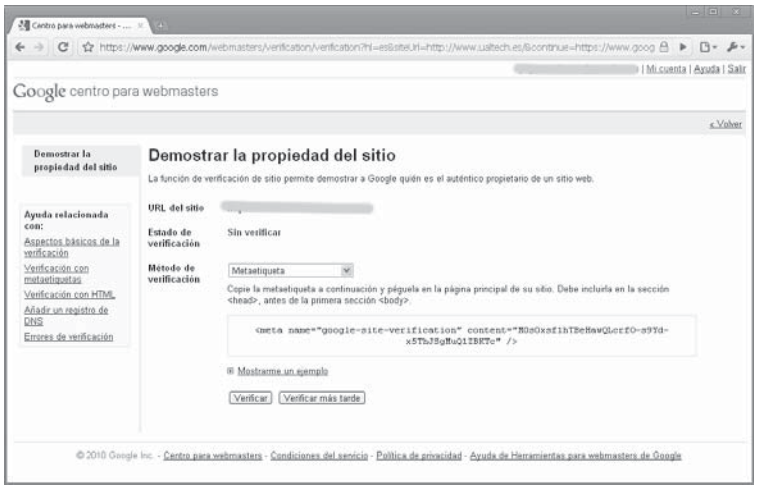


Figura 6-19. Herramientas para webmaster – Demostrando la propiedad del sitio

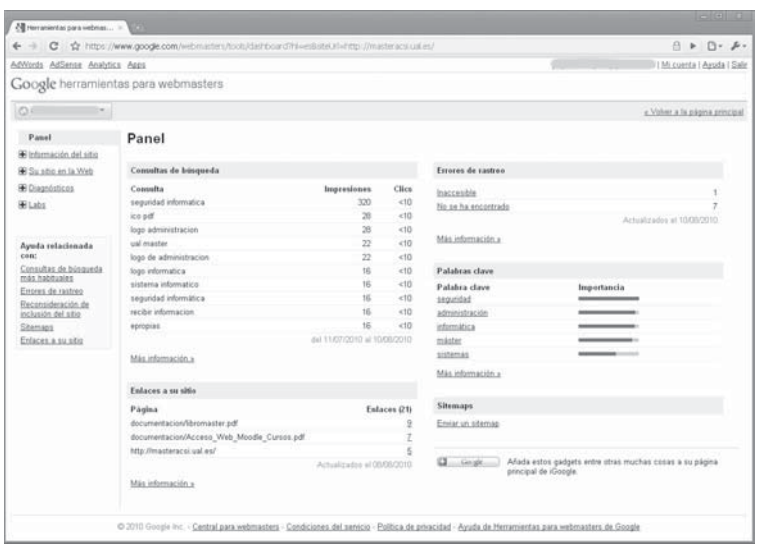


Figura 6-20. Herramientas para webmaster – Página principal

6.3.3 Google AdWords

Una forma muy efectiva de aumentar el número de visitas de nuestro sitio web es realizar una campaña de publicidad en Google. Las campañas de publicidad nos van a permitir poner anuncios de nuestro sitio para aumentar el número de visitas. Los anuncios que pongamos podrán aparecer en la página principal del buscador (véase la figura 6-21) o en las páginas colaboradoras que estén utilizando el servicio *AdSense*.

Figura 6-21. Anuncios en Google

Para gestionar las campañas de publicidad debemos darnos de alta en el servicio *AdWords* donde deberemos configurar nuestros datos personales y de facturación del servicio. Una vez dado de alta, deberemos crear una campaña de publicidad en Google. La campaña de publicidad está compuesta por anuncios y podemos tener todos los anuncios que estimemos oportunos.

Una vez creados los anuncios debemos dar de alta las palabras clave con las que son visibles nuestros anuncios. Para cada palabra clave deberemos realizar una puja. Es decir, tenemos que indicar cuánto dinero estamos dispuestos a pagar porque se vea nuestro anuncio.

De esta forma, cuando un usuario realiza una consulta en Google utilizando esa palabra clave se mostrarán los diferentes anuncios dependiendo de su valor de puja.

Para poder controlar el dinero que nos cuesta la campaña de publicidad resulta recomendable indicar el importe máximo diario que estamos dispuestos a pagar.

En la figura 6-22 podemos ver una captura donde se muestra información sobre una campaña de publicidad con sus respectivos anuncios y las palabras clave que se utilizan.

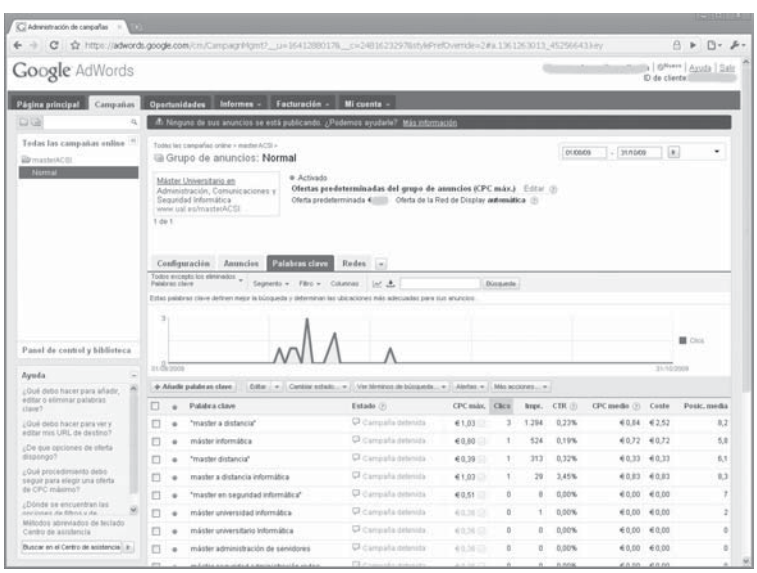


Figura 6-22. Google AdWords

Apéndice I

EL LENGUJAE SQL

I.1. Introducción

SQL (Structured Query Languaje) es el lenguaje estructurado estándar de consulta, definición, manipulación y control de bases de datos relacionales.

El lenguaje está formado por los siguientes elementos: comandos, cláusulas, operadores y funciones de agregado. Estos elementos se pueden combinar en las instrucciones para crear, actualizar y manipular las bases de datos.

A continuación se describen los principales elementos del lenguaje:

- **Comandos**. Instrucciones que identifican la acción a realizar con los datos. En la tabla I-1 podemos ver los principales comandos y operaciones.

Tabla I-1. Comandos u operaciones SQL

Método	Descripción
CREATE	Crea nuevas tablas, campos e índices.
DROP	Elimina tablas e índices.
ALTER	Permite modificar la estructura de las tablas (por ejemplo: agregando, eliminando o cambiando la definición de los campos).
SELECT	Se utiliza para extraer información de los registros de la base de datos.
INSERT	Permite insertar datos en la base de datos. Por ejemplo, permite insertar registros en una tabla.
UPDATE	Permite la actualización de registros de una tabla.
DELETE	Elimina registros de una tabla de una base de datos.

- **Cláusulas.** Permiten seleccionar los datos con los que vamos a realizar una operación. En la tabla I-2, podemos ver las cláusulas SQL.

Tabla I-2. Cláusulas SQL

Método	Descripción
FROM	Especifica la tabla o tablas de donde se seleccionan los registros
WHERE	Se utiliza para especificar los criterios de selección de registros
GROUP BY	Permite realizar grupos
HAVING	Especifica la condición que debe cumplir el grupo
ORDER BY	Permite ordenar los registros siguiendo un criterio (p.e. ordenados por nombre).

- **Operadores**. Son los elementos del lenguaje que permiten evaluar las operaciones de condición que deben cumplir los datos. Los operadores se utilizan en las cláusulas *WHERE* y *HAVING*. En la tabla I-3 se pueden ver los principales operadores de SQL.

Tabla I-3. Operadores

Método	Descripción
<	Menor que
>	Mayor que
<>	Distinto de
<=	Menor o igual que
>=	Mayor o igual que
=	Igual que
BETWEEN	Especifica un intervalo de valores.
IN	Permite especificar una lista a la que debe pertenecer el operando, en caso contrario devolvería falso
LIKE	Se utiliza para comparar cadenas de caracteres.
AND	Realiza la operación Y lógica
OR	Realiza la operación OY lógica
NOT	Negación lógica.
XOR	Realiza la operación OR Exclusiva

- **Operadores de agregación**. Se utilizan en la cláusula *SELECT* y permiten combinar registros, devolviendo la información en un solo registro. En la tabla I-4 podemos ver los operadores de agregación más utilizados.

Tabla I-4. Operadores de agregación

Método	Descripción
AVG	Calcula el promedio de los valores de un campo determinado
COUNT	Calcula el número de registros de la selección
SUM	Suma de todos los valores de un campo
MAX	Calcula el valor máximo de un campo
MIN	Calcula el valor mínimo de un campo
StDev	Calcula la desviación típica
Var	Calcula la varianza

Por funcionalidad podemos descomponer el lenguaje en tres categorías: sentencias de definición de datos (DDL), sentencias de modelado de datos (DML) y sentencias de control o seguridad de datos.

I.2. Definición de datos

Las sentencias de definición permiten definir las estructuras que almacenarán los datos en la base de datos. SQL dispone de los comandos CREATE, DROP y ALTER que permiten definir las tablas, vistas, procedimientos y triggers.

A continuación se pueden ver las principales instrucciones de definición de datos:

I.2.1. CREACIÓN Y ELIMINACIÓN DE UNA BASE DE DATOS

Prácticamente la creación de una base de datos consiste en la creación del conjunto de tablas que la componen, por ello, el estándar no posee una instrucción específica para la creación de la base de datos.

Algunos gestores de base de datos, como es el caso de MySQL, poseen una instrucción para la creación de una base de datos, cuya sintaxis es la siguiente:

```
CREATE DATABASE nombreBaseDatos;
```

Si lo deseamos, podemos crear un esquema para agrupar un conjunto de elementos. El estándar suministra la instrucción *CREATE SCHEMA* que permite crear un espacio de nombres para un conjunto de tablas, su sintaxis es la siguiente:

```
CREATE SCHEMA
       {[nombreEsquema]} | [AUTHORIZATION usuario]}
       [ElementosDelEsquema];
```

donde *nombreEsquema* es el nombre del espacio, *usuario* es el propietario del conjunto de tablas, vistas… y *elementosDelEsquema* son los diferentes elementos que contiene el esquema.

De igual forma se puede eliminar una base de datos o espacio de nombres, utilizando la instrucción *DROP SCHEMA* o *DROP DATABASE*.

```
DROP DATABASE nombreBaseDatos;
```

I.2.2. CREACIÓN Y ELIMINACIÓN DE UNA TABLA

Para crear una tabla se utiliza la sentencia *CREATE TABLE* donde debemos especificar el nombre y tipo de los campos de la tabla. La sintaxis de la instrucción es:

```
CREATE TABLE nombreTabla
( campo..
 [restriccionesTabla]
);
```

donde:

- *nombreTabla*. Es el nombre de la tabla.
- *Campoa*. Permite especificar el nombre y tipo de datos de cada campo así como sus principales características (p.e. longitud, si el campo puede ser nulo).
- *restriccionesTabla*. Permite crear las restricciones de la tabla.

Tabla I-5. Tipos de datos

ENTEROS	INT INTEGER TINYINT SMALLINT MEDIUMINT BIGINT UNSIGNED BIG INT INT2 INT8
TEXTO	CHARACTER(20) VARCHAR(255) VARYING CHARACTER(255) NCHAR(55) NATIVE CHARACTER(70) NVARCHAR(100) TEXT
REALES	REAL DOUBLE DOUBLE PRECISION FLOAT
NUMÉRICOS	NUMERIC DECIMAL(10,5) BOOLEAN DATE DATETIME

Tabla I-6. Restricciones de columna

Restricción	Descripción
NOT NULL	La columna debe obtener un valor no nulo.
UNIQUE	Evita que se produzcan valores repetidos en una columna
DEFAULT	Se establece el valor por defecto para la columna, si no se indica ninguno
CHECK	Permite realizar una comprobación antes de rellenar la columna
PRIMARY KEY	Indica que la columna es la clave primaria

Tabla I-7. Restricciones de tabla

Restricción	Descripción
UNIQUE	Conjunto de columnas que no pueden tener valores repetidos
CHECK	Permite especificar las condiciones que debe cumplir la tabla
PRIMARY KEY	Permite indicar el conjunto de columnas que forman la clave primaria
FOREIGN KEY	Indica que el conjunto de columnas especificadas es clave foránea que referencia la clave primaria especificada por el conjunto de columnas de la tabla dada

Por ejemplo, a continuación se está creando la tabla vehículo con los campos *modelo, motor, color, combustible, foto, precio y marca*:

```
CREATE TABLE vehículo (Modelo VARCHAR(20), Motor VARCHAR(20), Color
VARCHAR(20), Combustible VARCHAR(20), Foto VARCHAR(20), Precio
VARCHAR(20), Marca VARCHAR(20));
```

Para modificar una tabla se utiliza la sentencia *ALTER TABLE*, en este caso se presenta una sintaxis un poco más compleja, puesto que se pueden realizar bastantes más operaciones (añadir, modificar o eliminar una columna o restricción de columna, y/o añadir o eliminar restricciones de tabla).

```
ALTER TABLE
{
   ADD COLUMN nombreColumna parámetros |
   ALTER COLUMN nombreColumna |
   DROP COLUMN nombreColumna {RESTRICT|CASCADE}
}
|
{
   ADD restricción |
   DROP CONSTRAINT restriccionTabla {RESTRICT|CASCADE}
}
```

Para borrar una tabla se utiliza la sentencia *DROP TABLE*. La sintaxis es bastante sencilla ya que sólo admite dos parámetros: *RESTRICT* que impide borrar la tabla si está referenciada y *CASCADE* que borra la tabla y todo lo que referencie a la tabla.

```
DROP TABLE nombreTabla {RESTRICT|CASCADE};
```

I.3. Manipulación de datos

Son las instrucciones que permiten consultar, actualizar, añadir y eliminar registros de una base de datos.

A continuación vamos a ver un ejemplo de las principales operaciones con tablas, selección, inserción, actualización y eliminación.

I.3.1. CONSULTAS DE SELECCIÓN

SELECT se utiliza para seleccionar datos de una tabla. A continuación se puede ver su sintaxis.

```
SELECT nombreColumna1, nombreColumna2,…, nombreColumnaN | *
FROM nombreTabla1, nombreTabla2,… nombreTablaN
WHERE condicionesSelección;
```

Para el ejemplo, se va a suponer que tenemos una tabla *Agenda*, con los siguientes campos: *teléfono, nombre, dirección y correo*.

En el ejemplo siguiente, se realiza una consulta de selección que obtendrá todos los registros de la tabla cuyo nombre empiece por '*Jose'*. Los registros se mostrarán ordenados de forma descendente por el campo *número de teléfono*.

```
SELECT *
FROM agenda
WHERE nombre='Jose' ORDER BY telefono DESC
```

En el siguiente ejemplo, la consulta haciendo uso del predicado LIKE, mostrará todas las filas cuyo campo *nombre* empiece por 'J'.

```
SELECT *
FROM agenda
WHERE nombre LIKE 'J%'
```

I.3.2. CONSULTAS DE INSERCIÓN

INSERT se utiliza para insertar un nuevo registro en la tabla. Su sintaxis es la siguiente:

```
INSERT
INTO nombreTabla (nombreColumna1, nombreColumna2,…, nombreColumnaN)
VALUES (valorColumna1, valorColumna2,…, valorColumnaN)
```

Con la siguiente consulta se añade un nuevo registro a la agenda con los siguientes *datos: telefono=123456789, nombre=Antonio, dirección=C. Prueba Nº 12* y *email=antonio@123.es.*

```
INSERT
INTO agenda (telefono, nombre, dirección, email)
VALUES (123456789, "Antonio", "C. Prueba N° 12", "antonio@123.es")
```

I.3.3. CONSULTAS DE MODIFICACIÓN

Las consultas de modificación se realizan con el operador *UPDATE* que permite modificar uno o varios registros. Su sintaxis es:

```
UPDATE nombreTabla
SET nombreColumna=Valor
WHERE condicionActualizacion
```

En este caso, se debe especificar el campo o campos a actualizar y sobre qué fila o filas de la tabla se aplica la actualización. Por ejemplo, si queremos actualizar el valor del campo *dirección del registro* cuyo campo *teléfono* es *123456789* ejecutaremos:

```
UPDATE agenda
SET direccion = "Calle Nueva N° 23"
WHERE telefono=123456789
```

I.3.4. CONSULTAS DE ELIMINACIÓN

DELETE permite eliminar uno o varios registros de la tabla que cumplan una determinada condición. Su sintaxis es:

```
DELETE
FROM nombreTabla
WHERE condicionEliminacion
```

Por ejemplo, la siguiente consulta permite eliminar los registros de la tabla cuyo campo *teléfono* es *123456789*.

```
DELETE
FROM agenda
WHERE telefono = 123456789
```

Mientras que la siguiente elimina todos los registros cuyo campo *nombre* sea igual a '*Jose'*.

```
DELETE
FROM agenda
WHERE nombre LIKE 'Jose'
```

Apéndice II

HERRAMIENTAS Y URL REFERENCIADAS

A continuación, se muestran las herramientas y las páginas web referenciadas a lo largo del libro.

Tabla II-1. Herramientas referenciadas

Nombre	Tipo	URL
Apache	Servidor Web	http://www.apache.org/
Eclipse	Entorno de programación	http://www.eclipse.org/
Eclipse Web Developer Tools	*Plugin* de Eclipse para el desarrollo de portales Web	http://download.eclipse.org/releases/galileo (repositorio)
Google AdWords	Realizar campañas de publicidad en Google	https://adwords.google.es/
Google Analytics	Recopilación, visualización y análisis del tráfico de sitios Web	https://www.google.com/analytics
Google Keyword Tool	Selección de las mejores palabras clave para un sitio Web	https://adwords.google.es/select/KeywordToolExternal
Google Webmasters Tools	Análisis de visitas e indexación de sitios Web	https://www.google.com/webmasters/tools/
Joomla!	Gestor de contenido de tipo portales Web	http://www.joomlaspanish.org/
MediaWiki	Gestor de contenido de tipo *wiki*	http://www.mediawiki.org/
Moodle	Gestor de contenido de tipo *e-learning*	http://moodle.org/
MySQL	Sistema de gestión de bases de datos	http://www.mysql.com

MySQL Workbench	Diseño y modelado de bases de datos MySQL	http://wb.mysql.com/
MySQL-Front	Gestión y administración de bases de datos MySQL	http://www.mysqlfront.de/wp/
PHP	Lenguaje de programación para la creación de sitios Web dinámicos	http://www.php.net/
PhpMyAdmin	Interfaz Web PHP para la manipulación de bases de datos MySQL	http://www.phpmyadmin.net/
Search Engine Spider Simulator	Simulador del spider de un motor de búsqueda	http://webconfs.com/search-engine-spider-simulator.php
WordPress	Gestor de contenido de tipo *blog/bitácora*	http://wordpress.org/

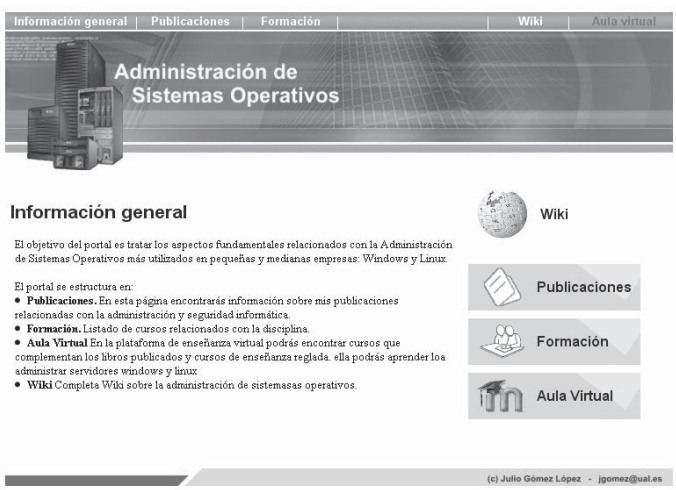
Tabla II-2. URL referenciadas

URL	Descripción
http://es.wikipedia.org/wiki/Tim_Berners-Lee	Biografía de Tim Berners-Lee
http://tools.ietf.org/html/rfc1738	RFC para URL
http://en.wikipedia.org/wiki/Web_colors#Web-safe_colors	Colores seguros para la Web (I)
http://www.colourlovers.com/	Colores seguros para la Web (II)
http://layouts.ironmyers.com/	Ejemplos de *layouts* para diseñar Webs
http://jigsaw.w3.org/css-validator/	Validador CSS del W3C
http://download.eclipse.org/releases/galileo	Repositorio con *plugins* para Eclipse
http://ayuda.joomlaspanish.org/ayuda-joomla/	Manual completo de Joomla!
http://docs.moodle.org/en/Main_Page	Documentación oficial de Moodle
https://api.wordpress.org/secret-key/1.1/	Web para generar las claves únicas de autenticación para la encriptación de las *cookies* de WordPress

http://es.wikipedia.org	Wikipedia, la enciclopedia universal
http://en.wikipedia.org/wiki/List_of_zoneinfo_time_zones	Información sobre los diferentes husos horarios
http://blog.nielsen.com/nielsenwire/online_mobile/top-u-s-online-search-providers-november-2009/	Top 10 de los buscadores más utilizados
http://www.mipagerank.com	Web para comprobar el PageRank de un sitio Web
http://www.myipneighbors.com/	Web para comprobar los dominios con los que compartimos hosting
http://www.seoprofesional.com	Blog sobre posicionamiento (I)
http://www.google.dirson.com	Blog sobre posicionamiento (II)
http://seoblog.es	Blog sobre posicionamiento (III)
http://www.google.com/intl/es/addurl.html	URL para dar de alta un sitio en el buscador Google
http://siteexplorer.search.yahoo.com/es/free/submit	URL para dar de alta un sitio en el buscador Yahoo!
http://www.bing.com/webmaster/SubmitSitePage.aspx	URL para dar de alta un sitio en el buscador Bing

PÁGINA WEB

Este libro dispone de su propia página web:



www.adminso.es

A través de esta página web los usuarios que se registren, podrán acceder a un curso virtual realizado con moodle. Para poder acceder al curso, el sistema le pedirá la contraseña:

POSICIONAMIENTO

El curso, que complementa la obra, le permite ampliar sus conocimientos a través de software referenciado a lo largo de la obra, conjuntos de transparencias, herramientas de autoevaluación para evaluar sus conocimientos, etc.

ÍNDICE ALFABÉTICO

A

Apache 80

B

Body 22
Break 115
Buscador 202

C

Caracteres especiales 25
Cascada 50
Clases 45
Colores 28, 55
Consulta SQL 121
Continue 115
CSS 41

D

Declaración de hojas de estilo 43
Do while 115
DTD 19

E

Eclipse 74
Elemento flotante 65
Enlace 30
Esquemas 69
Etiqueta 18
Eventos 36

F

For 114
Formulario 32
Frame 35
Funciones 116

G

Gestores de contenido 15, 133
Google AdWords 224
Google Analytics 214

H

Head 20
Herencia 49
Herramientas para webmasters 223
Hojas de estilo 14, 39
HTML 14, 17, 41

I

If 112

J

JavaScript 14
Joomla! 138

L

LAMP 79
Layouts 69
Listas 25

M

Media-types 56
MediaWiki 172
Modelo de caja 44
Moodle 154
MySQL 85, 120
MySQL Front 96
MySQL Workbench 91

O

Operadores 108

P

Page rank 204
Palabra clave 203, 207
PHP 14, 99, 103
PhpMyAdmin 90
Posicionamiento web 15, 201
Pseudoclases 46

R

Robots.txt 211

S

Selectores 50
Servidor de aplicaciones 79
Servidor de base de datos 85
Servidor web 80
Sitemap 209
SQL 15, 227
Switch 113

T

Tablas 26
Tipos de datos 106

U

Unidades 54
URL 31

V

Valores 53

W

While 115
WordPress 186

X

XHTML 41